Mathieu S. Capcarrere
Alex A. Freitas
Peter J. Bentley
Colin G. Johnson
Jon Timmis (Eds.)

LNAI 3630

Advances in Artificial Life

**8th European Conference, ECAL 2005
Canterbury, UK, September 2005
Proceedings**

ECAL 2005
European Conference on Artificial Life

Springer

Lecture Notes in Artificial Intelligence 3630

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Mathieu S. Capcarrere Alex A. Freitas
Peter J. Bentley Colin G. Johnson
Jon Timmis (Eds.)

Advances in Artificial Life

8th European Conference, ECAL 2005
Canterbury, UK, September 5-9, 2005
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Mathieu S. Capcarrere
Alex A. Freitas
Colin G. Johnson
University of Kent, Computing Laboratory
Canterbury CT2 7NF, UK
E-mail: {m.capcarrere, a.a.freitas, c.g.johnson}@kent.ac.uk

Peter J. Bentley
University College London, Department of Computer Science
Gower Street, London WC1E 6BT, UK
E-mail: p.bentley@cs.ucl.ac.uk

Jon Timmis
University of York, Departments of Electronic and Computer Science
Heslington, York Yo10 5DD, UK
E-mail: jtimmis@cs.york.ac.uk

Library of Congress Control Number: 2005931598

CR Subject Classification (1998): I.2, J.3, F.1.1-2, G.2, H.5, I.5, J.4, J.6

ISSN 0302-9743
ISBN-10 3-540-28848-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-28848-0 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11553090 06/3142 5 4 3 2 1 0

Preface

The Artificial Life term appeared more than 20 years ago in a small corner of New Mexico, USA. Since then the area has developed dramatically, many researchers joining enthusiastically and research groups sprouting everywhere. This frenetic activity led to the emergence of several strands that are now established fields in themselves. We are now reaching a stage that one may describe as maturer: with more rigour, more benchmarks, more results, more stringent acceptance criteria, more applications, in brief, more sound science. This, which is the natural path of all new areas, comes at a price, however. A certain enthusiasm, a certain adventurousness from the early years is fading and may have been lost on the way. The field has become more reasonable. To counterbalance this and to encourage lively discussions, a conceptual track, where papers were judged on criteria like importance and/or novelty of the concepts proposed rather than the experimental/theoretical results, has been introduced this year.

A conference on a theme as broad as Artificial Life is bound to be very diverse, but a few tendencies emerged. First, fields like 'Robotics and Autonomous Agents' or 'Evolutionary Computation' are still extremely active and keep on bringing a wealth of results to the A-Life community. Even there, however, new tendencies appear, like collective robotics, and more specifically self-assembling robotics, which represent now a large subsection. Second, new areas appear. 'Morphogenesis and Development' which used to be the subject of only a few papers, is now one of the largest subsections, and seems to be on the brink of becoming a field of its own. Finally, most classical themes of A-Life research like 'Artificial Chemistry', 'Ant-Inspired Systems', 'Cellular Automata', 'Self-Replication', 'Social Simulations' or 'Bio-realist Simulations' are still going strong and are well represented within this volume.

The conference this year has proven a great success with exactly 150 submissions, which is an all time high. This has allowed the programme committee to be fairly selective in its choice with only 74 papers accepted for full publication (49.3%). To avoid delaying the diffusion of novel ideas contained in works that were either less mature but promising, or controversial, a further 20 papers (13.3%) will be presented as posters but are published in full in these proceedings. The final selection by the organizing committee was greatly helped by the great professionalism of the programme committee. More than 95% of the reviews were done in time, and thus, all papers received at least 2 reviews with more than 88% of them receiving 3. Each paper that happened to be controversial was re-reviewed by the organizing committee and its acceptance or rejection decided individually.

Finally, the 'E' of ECAL stands for 'European', but this adjective, a legacy from its origin, remains true only in terms of the geographical location of the conference itself. We received papers from more than 40 countries, from Japan

to Brazil, from Norway to Australia, from Russia to China. This is great news not only for the conference but for the vitality of the field, and this great cultural mix will prove very fruitful at the conference.

To finish this preface, we would like to thank all the people who helped to organize ECAL 2005, and in particular, the members of the programme committee, the secretaries of the computing laboratory, Kate Friends and Jeanny Oatley and the webmaster, Andy Secker.

June 2005

Mathieu Capcarrere
Alex A. Freitas
Peter J. Bentley
Colin G. Johnson
Jon Timmis

Committees

Executive Committee

Conference chair:	Mathieu Capcarrere (University of Kent, UK)
Program chair:	Alex A. Freitas (University of Kent, UK)
Co-chairs:	Peter J. Bentley, Mathieu Capcarrere, Colin G. Johnson, Jon Timmis
Local Chair:	Jon Timmis (University of York, UK)
Workshops:	Peter J. Bentley (Univ. College London, UK)
Tutorials:	Colin G. Johnson (University of Kent, UK)

Programme Committee

Dr. Hussein Abbas, University of New South Wales, Australia
Prof. Andrew Adamatzky, University of the West of England, UK
Dr. Uwe Aickelin, University of Nottingham, UK
Prof. Nils A. Baas, University of Trondheim, Norway
Prof. Dr. Wolfgang Banzhaf, Memorial University of Newfoundland, Canada
Prof. Mark Bedau, Reed College, USA
Prof. Randall Beer, Case Western Reserve University, USA
Dr. Peter Bentley, University College London, UK
Prof. Hugues Bersini, Université Libres de Bruxelles, Belgium
Dr. Mark Bishop, Goldsmith College, UK
Dr. Tim Blackwell, Goldsmith College, UK
Dr. Henry Brighton, Max Planck Institute for Human Dev't, Berlin, Germany
Mr. Chris Buckley, University of Leeds, UK
Dr. Larry Bull, University of the West of England, UK
Dr. Seth Bullock, University of Leeds, UK
Dr. Mathieu Capcarrere, University of Kent, UK
Dr. Andre Carvalho, USP Sao Carlos, Brazil
Prof. Leandro de Castro, UniSantos, Brazil
Dr. Christopher D. Clack, University College London, UK
Dr. Pierre Collet, Université du Littoral, France
Dr. Son K Dao, HRL, USA
Prof. Kerstin Dautenhahn, University of Hertfordshire, UK
Mr. Simon Davy, University of Leeds, UK
Dr. Myriam Delgado, CEFET-PR, Brazil
Dr. Ezequiel Di Paolo, University of Sussex, UK
Prof. Marco Dorigo, Université Libre de Bruxelles, Belgium
Dr. Alan Dorin, Monash Univeristy, Australia
Prof. Dario Floreano, Swiss Federal Inst. of Tech., Lausanne, Switzerland

Dr. Alex Freitas, University of Kent, UK
Dr. Ivan Garibay, University of Central Florida, USA
Mr. Carlos Gershenson, Vrije Universiteit Brussel, Belgium
Prof. Inman Harvey, University of Sussex, UK
Prof. Takashi Hashimoto, Japan Advanced Institute of Science and Tech., Japan
Mr. Martin Hemberg, Imperial College, UK
Dr. Owen Holland, University of Essex, UK
Dr. Gregory Hornby, NASA Ames Research Center, USA
Prof. Phil Husbands, University of Sussex, UK
Prof. Auke Jan Ijspeert, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Prof. Takashi Ikegami, University of Tokyo, Japan
Dr. Colin Johnson, University of Kent, UK
Dr. Tatiana Kalganova, Brunel University, UK
Prof. Jozef Kelemen, Silesian University, Czech Republic
Dr. Sanjeev Kumar, George Mason University, USA
Dr. Laurent Lehmann, University of Cambridge, UK
Dr. Tom Lenaerts, Université Libre de Bruxelles, Belgium
Dr. John Levine, University of Edinburgh, UK
Dr. Heitor Lopes, CEFET-PR, Brazil
Mr. Robert Mach, ETH-Zurich, Switzerland, UK
Dr. Paul Marrow, BT Exact, UK
Dr. James Marshall, Bristol University, UK
Prof. Ian Marshall, University of Kent, UK
Prof. Alcherio Martinoli, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Dr. Claudio Mattiussi, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Prof. Jon McCormack, Monash University, Australia
Dr. Barry McMullin, Dublin City University, Ireland
Prof. Chris Melhuish, University of the West of England, UK
Prof. J.J. Merelo, Universidad de Granada, Spain
Dr. Martin Middendorf, University of Leipzig, Gemany
Dr. Julian Miller, University of York, UK
Prof. Alvaro Moreno-Bergareche, University of the Basque Country, Spain
Dr. Pablo Moscato, The University of Newcastle, Australia
Dr. Slawomir Nasuto, University of Reading, UK
Dr. Mark Neal, University of Wales, UK
Prof. Chrystopher Nehaniv, University of Hertfordshire, UK
Dr. Julio Cesar Nievola, PUC-PR, Brazil
Prof. Stefano Nolfi, CNR, Italy
Dr. Charles Ofria, Michigan State University, USA
Dr. Tim Otter, Crowley Davis Research, USA
Dr. Marco Pacheco, PUC-Rio, Brazil
Prof. Andres Perez-Uribe, Ecole d'Ingénieurs du Canton de Vaud, Switzerland
Dr. Carlos-Andres Pena-Reyes, Novartis Information and Knowl'ge Eng., Switz
Prof. Rolf Pfeifer, University of Zurich, Switzerland
Dr. Daniel Polani, University of Hertfordshire, UK

Prof. Riccardo Poli, University of Essex, UK
Prof. James Reggia, University of Maryland, USA
Mr. Ludovic Righetti, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Dr. Katya Rodriguez-Vazquez, National Autonomous Univ. of Mexico, Mexico
Prof. Eytan Ruppin, Tel Aviv University, Israel
Prof. Frank Schweitzer, ETH Zurich, Switzerland
Dr. Ana Sendova-Franks, University of the West of England, UK
Prof. Qiang Shen, University of Wales at Aberystwyth, UK
Prof. Moshe Sipper, Ben-Gurion University, Israel
Dr. George Smith, University of East Anglia, UK
Prof. Russell Standish, University of New South Wales, Australia
Dr. Andre Stauffer, EPFL, Switzerland
Prof. Susan Stepney, University of York, UK
Dr. Hideaki Suzuki, ATR Network Informatics Labs., Japan
Dr. Tim Taylor, University of Edinburgh, UK
Prof. Gianluca Tempesti, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Dr. Christof Teuscher, University of California San Diego, USA
Dr. Yann Thoma, Swiss Federal Inst. of Tech., Lausanne, Switzerland
Dr. Jon Timmis, University of York, UK
Dr. Peter Todd, Max Planck Institute for Human Dev't, Berlin, Germany
Prof. Marco Tomassini, Université de Lausanne, Switzerland
Prof. Andy Tyrrell, University of York, UK
Mr. Diemo Urbig, Humboldt-Universität zu Berlin, Germany
Dr. Andrew Watkins, Mississippi State University, USA
Prof. Claus Wilke, Keck Graduate Institute, USA
Mr. Hywel Williams, University of Leeds, UK
Prof. Andrew Wuensche, Discrete Dynamics Lab, USA
Prof. Fernando J. von Zuben, UNICAMP, Brazil

Table of Contents

Conceptual Track

Morphogenesis and Development

Robotics and Autonomous Agents

Evolutionary Computation and Theory

Cellular Automata

Models of Biological Systems and Their Applications

Ant Colony and Swarm Systems

Evolution of Communication

Simulation of Social Interactions

Self-replication

Artificial Chemistry

Posters

Effect of Synthetic Emotions on Agents' Learning Speed and Their Survivability

Šarūnas Raudys

Vilnius Gediminas Technical University, Saulėtekio 11, Vilnius, Lithuania
raudys@ktl.mii.lt

Abstract. The paper considers supervised learning algorithm of nonlinear perceptron with dynamic targets adjustment which assists in faster learning and cognition. A difference between targets of the perceptron corresponding to objects of the first and second categories is associated with stimulation strength. A feedback chain that controls the difference between targets is interpreted as synthetic emotions. In a population of artificial agents that ought to learn similar pattern classification tasks, presence of the emotions helps a larger fraction of the agents to survive. We found that optimal level of synthetic emotions depends on difficulty of the pattern recognition task and requirements to learning quality and confirm Yerkes-Dodson law found in psychology.

Keywords: synthetic emotions, bio-inspired modeling, multi-agent systems, cognition, learning, neural networks, Yerkes–Dodson law.

1 Introduction

It is commonly admitted in psychology that emotions are an evolutionary mechanism important for learning and survival, adaptation, perception, evaluation, reasoning, memory and decision making [1- 5]. Simililar conclusions have been obtained while investigating "computer emotions" [6-9]. Some computer scientists claim that feedback chains could be interpreted as synthetic emotions. So far most work was performed to analyse symbolic reasoning based algorithms. It was found that feedback chains could be useful for faster learning and cognition [8-9].

One may hope that training speed of connectionist based learning systems is also affected by synthetic emotions. There exists a large amount of neural networks literature (mostly dating from the late 80s and early 90s) which addresses rates of learning with backpropagation algorithm. The speed of the algorithm can be significantly enhanced by using adaptive learning rate, η, and momentum [10- 12]. Dynamical change of η is widely used to control back propagation training process of multilayer perceptron (MLP) (see e.g. [13]). The training speed, a type of the classification rule obtained and generalization error can be affected also by input data scaling, preliminary data transformations and a noise injection to training input data (see e.g. reviews [14, 15, Chapter 4]). For a general introduction into artificial neural networks, training speed and a generalization error problems see e.g. [15 -17].

In [18] it was demonstrated that training speed of the single layer perceptron (SLP) based classifier depends on difference $s = |t_1 - t_2|$ between desired outputs, t_1, t_2,

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 1 – 10, 2005.

corresponding to two diverse pattern classes. The training speed was measured as a number of training epochs required to train SLP. It was shown that while varying difference *s*, called stimulation, from 0 to 1 (if sigmoid activation function is used) the number of training epochs required to achieve a priori fixed classification error, P_{goal}, decreases at first, saturates and then starts increasing. It is Yerkes-Dodson (YD) law found in psychology [19, 20].

Up to now in psychology there is no unique definition of emotions. Significant role of emotions in evolution and adaptation suggests that there must be more than one mechanism for generating them [4]. From a variety of possible definitions of emotions, in present paper we relate synthetic emotions with training speed of the single layer perceptron used to solve pattern recognition task. We extend dynamic parameters' adjustment to target values and *associate synthetic emotions with dynamic changes in s, the stimulation*. We suppose that organization of the feedback chain assists in an increase in the stimulation if supervised learning was successful. We assume also that the feedback decreases the stimulation if learning was unsuccessful. This process may be called self-stimulation, reinforcement [16].

Our objective is not to investigate ways how to increase learning speed of back propagation algorithm. Our aim is to reveal principal mechanisms of interpretation of self-stimulation as synthetic emotions in connectionist learning systems. Global models could help in understanding factors affecting learning process in humans, societies and machines. They could offer to cognitive psychologists, sociologists and computer scientists one more model. In future they would assist in creating a variety of imitation situations for detailed studies of human and animal behaviors, improving strategies and algorithms to train single robots and groups of them.

To achieve this goal we look for the simplest model as possible. We selected a *nonlinear* SLP and a gradient descent supervised learning algorithm. In spite of simplicity of the mathematical model, in point of fact, a role of synthetic emotions on efficiency of connectionist learning systems was not considered so far.

The paper is organized as follows. Section 2 gives main terms and notations used in SLP training. In Section 3 we consider training algorithm with the feedback chain from a point of view of synthetic emotions: if classification error is decreasing, the stimulation level is increased, if classification error is increasing, stimulation is reduced. We show that such simple feedback chain, the synthetic emotions, support faster training. The dependence of training speed on level of emotions can be described by Yerkes–Dodson law [19, 20] too. Section 4 considers a situation where there are a large number of similar artificial agents that ought to learn changing pattern classification tasks. We show that presence of emotions helps larger fractions of agents to survive, i.e. to learn to solve the task in *a priori* fixed number of training iterations. Section 5 contains discussion and suggestion for future research work.

2 Training Peculiarities of the Single Layer Perceptron

Classical approach in adaptive learning is rooted in psychology, going back to early work of Thorndike [21] on animal learning and that of Pavlov [22] on conditioning. Here learning takes place through a process of punishment and reward with the goal of achieving a highly skilled behavior. In artificial intelligence, the learning is

performed through continued interaction with the environment in order to minimize a scalar index of performance, a fitness (cost) function. To explain the principal trends, in present paper we start the analysis with the simplest model purposely. In order to explain a sense of our analysis, below we will present necessary definitions and show fundamental feature of gradient descent training: the weights of the single layer perceptron are increasing and gradually start slowing down the speed of training process. *The weights increase is principal importance and a novelty of our analysis*.

In our formulation, objects (situations) to be classified to one of the categories are described by input feature vectors, $\boldsymbol{x} = (x_1, x_1, \ldots x_p)$. The perceptron calculates a weighted sum of inputs, $sum = w_1 \times x_1 + w_2 \times x_2 + \ldots + w_p \times x_p + w_0$. A set of p values, w_1, $w_2, \ldots, w_p$, is called a weight vector, $\boldsymbol{w}$, and w_0 is a weight threshold value. We will use vector notation $sum = \boldsymbol{w} \times \boldsymbol{x} + w_0$. Very important essential of the perceptron is transfer function. Weighted sum, *sum*, is supplied to nonlinear element that calculates *output* of the perceptron as a non-linear function of *sum*. As an example one can consider sigmoid function, $output = 1/(1+\exp(-sum))$. If $sum = 0$, $output = 0.5$ (middle value). If *sum* is large negative, *output* is close to 0. If *sum* is large positive, *output* is close to 1. Note that a slope of function $output = f(sum)$ is the highest where $sum = 0$. If *sum* moves toward ± *infinity*, the slope diminishes and approaches zero [24, 25].

In order to use SLP practically one needs to know coefficients $w_0, w_1, \ldots, w_p$. To find the coefficients, we utilize training data called a training set, the vectors of the categories **A** and **B**: $\boldsymbol{x}_1^{(1)}, \boldsymbol{x}_2^{(1)}, \ldots, \boldsymbol{x}_{N_1}^{(1)}$ from **A** and $\boldsymbol{x}_1^{(1)}, \boldsymbol{x}_2^{(2)}, \ldots, \boldsymbol{x}_{N_2}^{(1)}$ from **B**. In perceptron training, we require that for class **A** *output* ought to be close to a priori selected target, t_1. For another class, **B**, we have to choose another value, e.g. $t_2 = 1 - t_1$ [14, 15, 23]. Traditional algorithm used to train SLP is back propagation, where usually a sum of squares cost function, *cost*, is minimized,

$$cost = 1/N \sum_{i=1}^{2} \sum_{j=1}^{N_i} \left(t_j^{(i)} - f(\boldsymbol{w} \times \boldsymbol{x}_j^{(i)} + w_0) \right)^2, \quad (1)$$

where $\boldsymbol{w}$ is unknown p-variate weight vector, w_0 is a bias term, both to be found during training process, $t_j^{(i)}$ is a desired output (a target) of the perceptron if vector $\boldsymbol{x}_j^{(i)}$ is presented to its input.

In this paper we consider symmetric targets, $t_j^{(1)} = 0.5 - 0.5s$, $t_j^{(2)} = 0.5 + 0.5s$, $(0 < s \leq 1)$. If parameter s is close to 0, we have similar target values. If s is close to 1, targets $t_j^{(1)}$ and $t_j^{(2)}$ are close to boundary values of sigmoid activation function $f(sum) = 1/(1+\exp(-sum))$, i.e. 0 and 1. Thus, parameter s is interpreted as *the strength of training signal*, *the stimulation*. During training (adaptation) process, new vector, $\boldsymbol{w}_{(t+1)}$, is equal to the previous one, $\boldsymbol{w}_{(t)}$, plus a correction term:

$$\boldsymbol{w}_{(t+1)} = \boldsymbol{w}_{(t)} + CT_{ij}^t, \quad (2)$$

where $CT_{ij}^t = -\eta \times (t_j^{(i)} - f(sum)) \times (\partial f(sum)/\partial sum) \times (\partial sum/\partial \boldsymbol{w})$ is the correction term, η is called learning rate (step) parameter, $t_j^{(i)} - f(sum)$ is an error signal – a difference

between the desired and actual outputs of the perceptron, $sum = \boldsymbol{w}' \boldsymbol{x}_j^{(i)} + w_0$, $\partial f(sum)/\partial sum$ is the derivative of the activation function and $(p+1)$-dimensional vector $(\partial f(sum)/\partial sum) \times (\partial sum/\partial \boldsymbol{w})$ is called a gradient.

If the weights are small, the gradient, $(\partial f(sum)/\partial sum) \times (\partial sum/\partial \boldsymbol{w})$, is large. Simple algebra shows that when the weights are large, the gradient becomes small. During training, the magnitudes of the weighs are increasing and affect properties of the cost function [15, 25]. Moreover, with an increase in the magnitude of the weights, the gradient is decreasing towards zero. It means that in situations when the agent (the perceptron) has learned to solve its task properly and the weights are already large, *due to the large weights the perceptron is unable to re-learn a new task quickly*.

Two parameters, the learning rate parameter, η, and a difference between desired outputs, t_1, t_2, of the perceptron, $s = |t_1 - t_2|$ can be utilized to control the training process. We are increasing η or s by multiplying/dividing these parameters by positive scalar γ if training was successful/unsuccessful during n_{inertia} training epochs. We interpret parameter γ as *self-stimulation* or *synthetic emotions*. We remind that the self stimulation model is only one definition of emotions from a variety of possible ones. We will show that parameter γ affects training speed. Training speed is measured by a number of training epochs required to achieve a goal, P_{goal}, an *a priori* defined classification performance.

3 Influence of Self-stimulation on Speed of Training Process

In order to investigate the feedback chains, we consider simple adaptation model and perform simulation studies utilizing uncomplicated data – two bi-variate Gaussian classes with mean vectors $\boldsymbol{\mu}_2 = -\boldsymbol{\mu}_1$, unit variances and correlation between the variables ρ=0.7. In this model, three parameters control η and s: multiplication factor γ, sensitivity parameter Δ ($\Delta > 1$) and delay (inertia) n_{inertia} after which correction of η or s is made. The parameter, γ, indicates relative increase or decrease of learning step, η, or stimulation parameter, s, if training was effective or ineffective during n_{inertia} training epochs. Let $\Psi = cost(t) / cost(t\text{-}n_{\text{inertia}})$ be a ratio of current cost (1) with previous cost value calculated n_{inertia} epochs before. Parameters η or s are multiplied by factor γ, if $\Psi < 1/\Delta$. The parameters η or s are divided by factor γ, if $\Psi > \Delta$. Otherwise, nothing is changed. In present paper, we report results obtained when $n_{\text{inertia}} = 1$ and Δ= 1.01. In analysis of dynamic learning step change, at start, we select initial learning step value, η_0.

Dynamic change of learning step, η. Results of the experiments with three P_{goal} values (0.003, 0.01, 0.03) and two starting learning step values, 0.1, and 125, are presented in Fig. 1 (stimulation s=1). Three graphs obtained for η_0=0.1 indicate that dynamic change of learning step, η, speeds up training process. In spite of the fact that we started training from small initial η value, feedback chain results that parameter η grows very quickly and compensates exponential decrease of the gradient caused by the gradual increase of the weights magnitudes.. If starting learning step is

small (e.g. η_0=0.1), at the very beginning we do not have notable decrease of the cost (1). Due to high value of sensitivity parameter (Δ=1.01) parameter η remains unchanged for a long time. Training process remains very slow. If starting η value is too large (e.g. η_0=125), immediately after the first iteration we have very large change of the weight vector and a saturation of the cost function. Learning becomes slow and unstable. In certain cases, learning even stops (graphs 1*b* and 2*b* in Fig. 3).

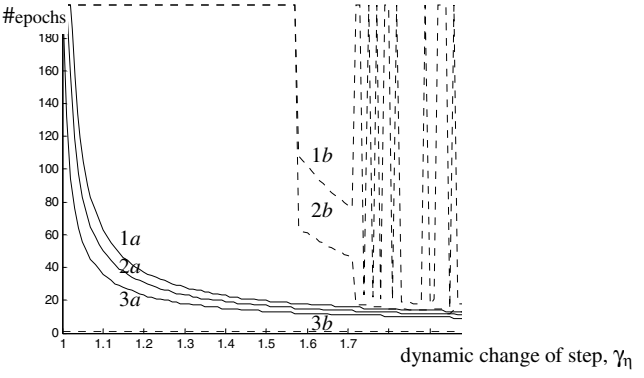


Fig. 1. A number of training epochs required to achieve P_{goal} as function of dynamic change of learning step: 1 - P_{goal} = 0.004; 2 - P_{goal} = 0.01; 3 - P_{goal} = 0.03. Almost non-overlapping classes, $\boldsymbol{\mu}_1$ = (0.6 2.4); $\boldsymbol{w}_{start}$ = [0.08 1 2]. Curves marked by "*a*" start training from η_0=0.1 (stable training) and that marked by "*b*" – start from η_0=125 (non-stable training).

We see that dynamic η change speeds up training process. In MLP training, it was used primary to help climb out from false local minimum [12, 13, 15, 17]. Dynamic η change is useful if *a priori* we do not know proper value of learning step parameter. Moderate dynamic η change assist in overcoming large weights effects if nonlinear soft-limiting activation function is used. We see also that sometimes dynamic η change could become dangerous. Therefore, we conclude that dynamic η change is suitable, however, sometimes imperfect model of synthetic emotions.

Dynamic change of stimulation, *s*. For the start, relatively small stimulation value (s_{start} = 0.002) was selected, i.e. t_1 = 0.499, t_2 = 0.501. In Fig. 2 we have a number of training epochs required to achieve P_{goal} as a function of parameter γ_s, dynamic change of stimulation strength (η=2.0; three values of P_{goal} and two values of starting weight vector, $\boldsymbol{w}_{start}$). In both experiments with different initial weights, position of starting decision boundary was the same, only magnitudes of the weights differed. To examine situations where large perceptron's weights start slowing down the training speed, almost non-overlapping pattern classes with small classification error were considered.

In second experiment, components of the initial weight vector, $\boldsymbol{w}_{start}$, w_0, were 1.5 times larger as in the first one. For that reason, at the very start we had larger sums, $sum = \boldsymbol{w}_{start} \times \boldsymbol{x}_j^{(i)} + w_0$. Consequently, the gradients turned out to be smaller at the very

start of training. Consequently the training process became slower. Graphs in Fig 2*ab* indicate that in training with larger initial weights, we need higher number of training epochs (the learning task becomes more difficult).

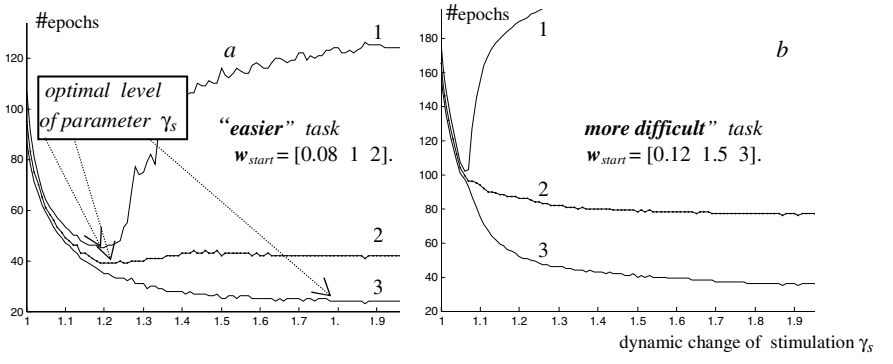


Fig. 2. A number of training epochs required to achieve P_{goal} as a function of dynamic change of stimulation strength, γ_s: 1 - P_{goal} = 0.004; 2 – P_{goal} = 0.01; 3 – P_{goal} = 0.03; Almost non-overlapping pattern classes, $\boldsymbol{\mu}_1 = (0.6\ 2.4)$; *a*) – $\boldsymbol{w}_{start} = [0.08\ 1\ 2]$, *b*) – $\boldsymbol{w}_{start} = [0.12\ 1.5\ 3]$.

Graphs in Fig. 2 demonstrate that higher requirements to *learning quality* (smaller values of P_{goal}) necessitate higher number of training epochs. Both families of the curves indicate that *for each requirement for learning quality there exists an optimal level of parameter* γ_s *where training is fastest*. Both larger initial weights and smaller P_{goal} increase the difficulty of the task. We pay readers attention that *in difficult tasks, optimal values of parameter* γ_s *are smaller*. *In easier tasks optimal level of parameter* γ_s is *higher*. If the classes overlap notably (the classification task is more difficult), classification errors prevent excessive growth of the weights. The weights are smaller and the minima of the curves $\#_{epochs} = f(\gamma_s)$ are less expressed. (Fig. 3).

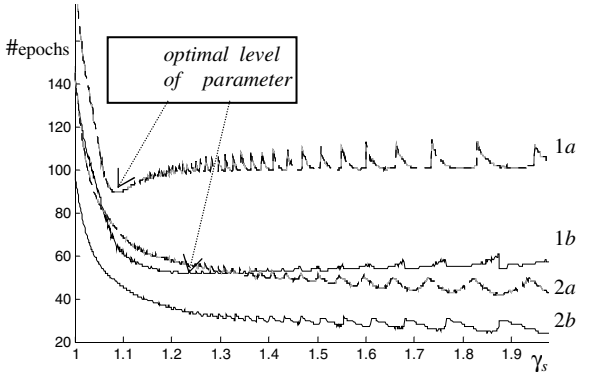


Fig. 3. A number of training epochs required to achieve P_{goal} as a function of dynamic change of stimulation, γ_s: 1 - P_{goal} = 0.08; 2 - P_{goal} = 0.16; two notably overlapping pattern classes: $\boldsymbol{\mu}_1$ = [0.3 1.2]: *a* - $\boldsymbol{w}_{start} = [0.12\ 1.5\ 3]$, *b* – $\boldsymbol{w}_{start} = [0.08\ 1\ 2]$.

4 Survivability of Population of Intellectual Agents

A natural question that arises while analyzing biological populations, social collectives and multi-agent systems in context of their evolution, is an influence of self-stimulation on survivability of the population. Consequently, we have to consider a situation where there are a large number of similar artificial agents that ought to learn new pattern classification tasks. *We assume that the agent passes away if it fails to learn fast enough to satisfy the a priori fixed condition* $P_{\text{classif}} < P_{\text{goal}}$ after the $t_{\max}$ *training epochs*. We show that presence of synthetic emotions helps larger fractions of agents to survive, i.e. to learn to solve the task rapidly. As a first step in the population analysis we will investigate many populations of agents having *different* self-stimulation parameter, γ_s. To have larger diversity of agents we assume that each population of agents is composed of r_f sub-families, f_r agents in each sub-family. Thus, in each population we have $r = f_r \times r_f$ agents. All agents in one sub-family possess *similar* characteristics, however, the families are to some extent *different*.

In experiments reported below, f_r =200*;* r_f = 6, r = 1200, the starting stimulation, s_{start} = 0.002, Δ = 1.01. We used two-dimensional Gaussian data with fixed mean vectors; $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_2 = [0.6\ 2.4]^T$; correlation ρ=0.7. Initial weights, learning step η, variances, σ_1, σ_2, of the single data components, however, were random variables: $\boldsymbol{w}_{start} = \tau \times [0.08\ 1\ 2]$, $\tau = (0.999+0.3\zeta)$, $\eta = 0.5+0.05\zeta$, σ_1, $\sigma_2 = 1+0.1\zeta$, where all ζ were independent random variables composed of sum of two random variables, ζ_A and ζ_B, distributed uniformly in interval [-0.15 0.15]. Variable ζ_A was individual for each single agent, while variable ζ_B was common to r_f agents in single sub-family.

In the Fig. 4*a* we have curves: a mean number of epochs required to achieve goal P_{goal} during $it_{\max}$ = 200 epochs versus self-stimulation parameter, γ_s. Curves 1 and 2 remind curves 1 and 3 in Fig. 2*a* obtained for one single agent. A main difference is a scale for the number of training epochs allowed to reach the goal. In Fig. 3, learning step was approximately four times smaller. Therefore, we needed approximately four times more epochs to achieve the goals.

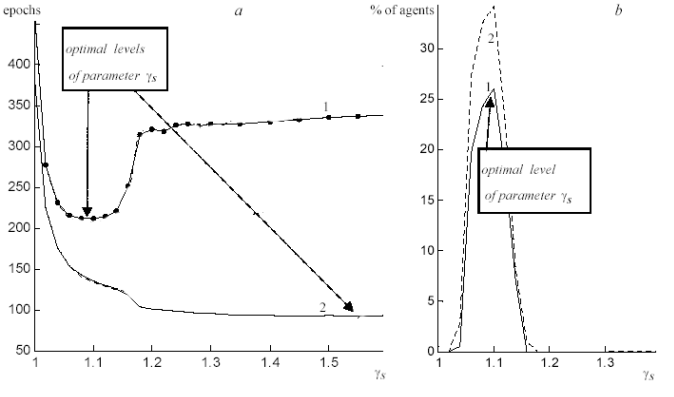


Fig. 4. *a*) Mean number of training epochs required to achieve P_{goal} as a function of dynamic change of self-stimulation, γ_s: 1 - P_{goal} = 0.004, 2 - P_{goal} = 0.03 during 200 training epochs. *b*) An effect of emotions ES on populations survivability (a fraction of agents which achieved desired classification error level, P_{goal}=0.004): 1 $it_{\max}$ = 200 epochs, 2 - $it_{\max}$ = 600 epochs.

In Fig. 4*b* we have "survivability graphs", a fraction of agents (in percents) that succeeded achieve the goal, $P_{goal} = 0.004$ during it_{max} training epochs. The graphs in 4*b* also demonstrate the Y-D Law and indicate an existence of optimal values of self-stimulation parameter, γ_s.

The graphs in Fig. 4*b* show that survival ratio crucially depends on a number of epochs allowed to achieve the goal, P_{goal}. If short training is available, in order to survive, optimal amount of emotions depends on P_{goal} notably. The stricter are requirements for learning quality (lower goal value), the smaller amount of emotions ES is necessary. For easier training tasks (high goal value, small initial weights), higher level of emotions is allowed.

5 Discussion and Suggestions for Future Research

One may assume that in process of species evolution, nature selected only these populations of individuals which are controlling the stimulation. This "self-stimulation phenomenon" we interpret as emotions in our study. To be more precise, it is only one aspect of emotions in this paper called "emotions ES" (emotions for stimulation). To imitate synthetic emotions we restricted ourselves with simplest model: the single layer perceptron trained by back propagation algorithm and feed-back chain utilized to modify stimulation values in dependence on a success in training. The new modification of the perceptron training algorithm adapts its target values during iterative training process: if classification error decreases for some time, the stimulation level is increased, if training becomes unsuccessful, the stimulation is reduced. Simulation studies demonstrate that values of feedback control of the targets affect the speed and performance of training process. Training scenario with emotions supports faster training. The dependence of training speed on level of emotions can be described by the Yerkes–Dodson law and has inverted letter "U" shape.

In analysis of situation where there is a large number of artificial agents that ought to learn different, subsequently changing pattern classification tasks, it was found that if short training is allowed, presence of certain amount of emotions helps larger fractions of agents to survive. This peculiarity of adaptive agent could be used in multi agent systems research.

Simulation analysis of the optimal level of self-stimulation on requirements to learning quality exhibit several features that match experimental findings in humans [1, 4, 5]. Often the people are changing their motivation in dependence on the success. Children are more emotional as the grown up persons. Populations which settle in warm, easy to live climate raise lower requirements to learning quality. So, they are more emotional than ones which live in severe northern climate. In this way, very simple model of dynamic target value control is one more possibility to elucidate theoretically these important observations. Possibly, our definition of synthetic emotions could indicate new ways to train robots, intellectual agents that have to adapt and to operate in unproblematic or more difficult environmental conditions. We have seen that dynamic change of learning step allows speed up training process notably. We investigated both, dynamic change of η and s. Dynamic target value control, however, is easier to interpret from a point of view of the emotions.

Striking conclusion derived from above analysis is that such simple element as single layer perceptron equipped with dynamic change of the difference between desired outputs could be interpreted as synthetic emotions which help the agents overcome difficulties that arise during training process. In the output layer of multilayer perceptrons, we also have SLPs trained by the same type of learning algorithm. One may hope that conclusions obtained for SLP classifier most likely are valid for MLP based learning systems. In principle, significantly larger number of means can be utilized to control perceptron's training process. These means include a noise injection to components of training vectors, a noise injection to desired targets, use of weight decay term for regularization, etc. [14, 15]. Obviously, these factors could be included into the emotions model to make it more suitable to analyze real world problems. Useful extension of the model is analysis of learning process utilized to solve not a single, but a variety of varying pattern recognition tasks, like it was done in a recent research papers on aging [25] and criminality [26]. In analysis of populations of intelligent agents, one can consider situations where successful members of the populations can produce offspring, a strategy widely widespread in Nature [26, 27]. Including of the factors just mentioned, organizing of the populations into sub-groups with mechanisms of self-support of the agents inside the sub-group and restricted beneficial cooperation between them, and incorporation of the mechanisms of synthetic emotions into learning rule is a topic for future research in order to crate more realistic multi agent systems capable to survive in permanently changing environments.

The investigation performed shows that connectionist approach to examine synthetic emotions can be effective enough to explain numerous phenomena observed in real life. Modifications of single layer perceptron based training model could be useful both for cognitive psychologists, sociologists, economists as a mean to investigate learning processes and to computer scientists which develop the algorithms capable adapt rapidly in unknown and everlastingly changing environments.

Acknowledgments

The author thanks Prof. Viktoras Justickis and Vanda Dovlias for useful and challenging discussions.

References

[1] Ortony, A., Clore, G., and Collins, A. (1988). *The Cogitive Structure of the Emotions*. New York: Cambridge University Press.
[2] Griffiths, P. (1997). *What Emotions Really Are: The Problem of Psychological Categories*. Chicago: Chicago University Press.
[3] Oatley, K., and Jenkins, J. (1996). *Understanding Emotions*. Oxford: Blackwell.
[4] Izard C.E. (1993) Four systems for emotion activation – cognitive and noncognitive processes. *Psychological Review*, 100 (1): 68-90.
[5] Lazarus, R.S. (1991). *Emotions and adaptation.* New York: Oxford University Press
[6] Goleman, D. *Emotional Intelligence: Why It Can Matter More than IQ*. London: Bloomsbury Publishing, 1996.

[7] Pfeifer, R. The "fungus eater approach" to emotion: a view from artificial intelligence. *Cognitive Studies, The Japanese Society for Cognitive Sci.,* 1:42-57, 1994.
[8] Chwelos G. and Oatley K. Appraisal, computational models, and Scherer expert system, *Cognition and Emotion*, 8(3): 245-257, 1994.
[9] Picard, R. W. *Affective Computing*. Cambridge: The MIT Press, 1997.
[10] Jacobs R. A. Increased rates of convergence through learning rate adaptation, *Neural Networks*, vol. 1(3): 295–307, 1988.
[11] Radi A. and Poli R. Genetic programming discovers efficient learning rules for the hidden and output layers of feedforward neural networks", In*: Genetic Programming, Proceedings of EuroGP'99"*, (Poli R and Nordin P. Eds.), Springer-Verlag, Vol. 1598 (26-27): 120-134, 1999.
[12] Yao X. Evolving artificial neural networks, *Proc. of IEEE*, 87(1): 1423-1447, 1999.
[13] [The MathWorks, *Matlab: The language of technical computing* (www.mathworks.com, 1998)
[14] Raudys S. Classifier's complexity control while training multilayer perceptrons. *Lecture Notes in Computer Science*. Springer-Verlag, 1876, 32-44, 2000.
[15] Raudys S. *Statistical and Neural Classifiers: An integrated approach to design*. Springer-Verlag. NY, 2001.
[16] Sutton R.S. and Barto A.G. *Reinforcement Learning: An introduction*. MIT Press,: Bradford Book, Cambridge, MA, 1998.
[17] Haykin S. *Neural Networks: A comprehensive foundation*. 2nd edition. Englewood Cliffs, NJ: Prentice-Hall, 1999.
[18] Raudys S. and Justickis V. Yerkes-Dodson law in agents' training. *Lecture Notes in Artificial Intelligence,* Springer-Verlag, Vol. 2902, pp. 54-58, 2003.
[19] Yerkes, R. M. and Dodson, J.D. The relation of strength of stimulus to rapidity of habit-formation. *Journal of Comparative Neurology and Psychology*, 18:459-482, 1908.
[20] Teigen K.H. Yerkes-Dodson – a law for all seasons. *Theory and Psychology*, vol. 4 (4): 525-547. 1994.
[21] Thorndike, E. L. *Animal Intelligence*. Darien, Connecticut: Hafner, 1911.
[22] Pavlov I. P. New researches on conditioned reflexes, *Science*, 58:359-361, 1923.
[23] Rumelhart D.E., Hinton G.E. and Williams R.J. Learning internal representations by error propagation. In: D.E. Rumelhart, J.L. McClelland (editors), *Parallel Distributed Processing: Explorations in the microstructure of cognition* vol. I, pp. 318–62, Cambridge, MA: Bradford Books, 1986.
[24] Raudys S. Evolution and generalization of a single neurone. I. SLP as seven statistical classifiers. *Neural Networks*, 11(2): 283-296, 1998.
[25] Raudys S. An adaptation model for simulation of aging process. *Int. J. of Modern Physiscs, C*. 13(8): 1075-1086, 2002.
[26] Raudys S., Hussain A., Justickis V., Pumputis A. and Augustinaitis A. Functional model of criminality: simulation study. Proc. CONTEXT'2005 (Turner R., ed.), *Lecture Notes in Computer Science,* Springer-Verlag, 2005 (in Press).
[27] Raudys S. Survival of intelligent agents in changing environments. *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 3070, 109-117, 2004.

From the Inside Looking Out: Self Extinguishing Perceptual Cues and the Constructed Worlds of Animats

Ian Macinnes and Ezequiel Di Paolo

Centre for Computational Neuroscience and Robotics,
John Maynard Smith Building, University of Sussex,
Falmer, Brighton, BN1 9QG
I.A.Macinnes@sussex.ac.uk, ezequiel@sussex.ac.uk

Abstract. Jakob von Uexküll's theory of the *Umwelt* is described and it is used to show how perceptual states can be defined. It is described how perceptual cues are selected over evolutionary time and defined by the organism that experiences them. It is then argued that by applying the model of the *Umwelt* to describe an animat's behaviour, we can model the normally distributed, dynamic activations of the animat as discrete perceptual states.

1 Introduction

This paper suggests we use a model to describe the behaviour of animats in terms of their constructed worlds and that by doing so, we stand to gain in two ways. The first way is that we find that have a model of the perceptual states of an animat, and so can apply various tools associated with state machines, including calculating the entropy of a constructed world. The second way is that by evolving the relationships between an animat and the objects in its environment that it interacts with, we are making the animat's constructed world more evolvable.

1.1 Controllers, Perceptions and the Constructed World

The nature of awareness and the mechanism of how perceptions are generated from physical matter has provoked much debate among philosophers. The most influential arguments regarding these subjects in the western world were made by the philosopher Immanuel Kant [1]. His arguments are complicated but among his conclusions is that the real nature of the world is unknowable, and we can only classify any possible external reality of an object through our perceptions of it rather than through direct knowledge of the object itself. Kant believed that the subjective sense of self is constructed through the process of the organisation

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 11–20, 2005.

of perceptions. Whether perceptions are epiphenomena or play a causal role in behaviour, it is commonly supposed that a constructed world's function as part of an organism is managing information about the organism's environment and physical self.

The idea of using parameters to represent perceptions, mental states, or bits of knowledge has been highly influential for the majority of work performed in artificial intelligence and the disadvantages of it are well known [2]. It is tempting to use explicit representational states because of their clear information content. It is an easy task to calculate attributes such as the entropy of a controller with states over a period of time, or see how the states produce behaviour that result in other states [3]. Information and its transmission has long been used to explain animal behaviour, cognition, and evolution. It is very difficult to use these state based methods to calculate the information within the animats we evolve in evolutionary robotics as we do not use the concept of states, but instead allow control to self-organise from underlying parts. How are we able to determine that a state exists, or succeeds another state if they are not explicit? How do we assign states to the varying activation of a distributed structure?

Presently we analyse such systems in terms of dynamic processes such as attractors, limit cycles, etc. [4], together with the analysis of the role of specific neurons. The analysis required to understand even simple systems can be complicated and it is not clear how general the conclusions made about one system can be applied to another. There has been work performed in the field of artificial life on classifying information within an animat's distributed dynamic controller. Research has been performed on attempting to measure and analyse representational activations within an embodied robots neural network controller in terms of information [5]. Research has also been performed on the application of information theory to measure the information flow through perception-action loops of robots [6]. However, much work in evolutionary robotics tries to explain behaviour and perceptual organisation by analysing the dynamic activations of structural parts of a controller.

The line of enquiry described in this paper takes a different approach. It describes a model of the constructed world as defined by the ethologist Jakob von Uexküll [7], and why we gain by applying it to describe the behaviour of the animats we evolve. His theory was heavily influenced by the constructivist ideas of Kant. If we use his theory to explain animat's behaviour, then we have a simplified model of the constructed world generated by the animat. It is hoped that applying von Uexküll's theory will help overcome the problem of defining information within complex self-organising controllers, allow us to improve the evolvability of our animats, and perhaps lead to insights regarding the constructed worlds of robots and organisms.

First we will look at von Uexküll's theory of the *Umwelt*. Then we will see how we can distinguish perceptual states in his model. Finally we will argue that we gain by applying the model of the *Umwelt* to the less representative methodologies used in the field of artificial life.

2 Negotiating Functional Circles and the *Umwelt*

2.1 Functional Circles and an Organism's Constructed World

Von Uexküll explained purposeful animal behaviour by linking the organism's *phenomenal world* (the world as perceived) and its *effector world* (the world as enacted) into a single closed whole, the *Umwelt*. The *Umwelt* as defined by von Uexküll consists of a set of functional circles (figure 1). A functional circle defines the interaction between an organism and an element or object within its environment. They are abstract structures that tie together a subjective experience or perception (termed a *perceptual cue*) and the effect that the perception has on the behaviour of the organism (called an *effector cue*). Von Uexküll used the theory to demonstrate that an organism doesn't respond to its environment, but rather to its perception of the environment. Functional circles provide a model of how an organism's perceptual world is continually constructed as part of the organism's ongoing interaction with its environment. Using von Uexküll's model of the *Umwelt*, all of an organism's knowledge of itself and its environment is ultimately constructed from the perceptions within the subjective world it generates.

Von Uexküll himself provided the example of the purposeful behaviour of a female tick [7], of which he described three functional circles. The tick waits on a twig until a mammal moves close to the tick. The tick then jumps onto the mammal and burrows around in the mammal's fur until the tick finds a suitable place on its skin to bite a hole to suck blood from. This sequence of actions can be described using functional circles.

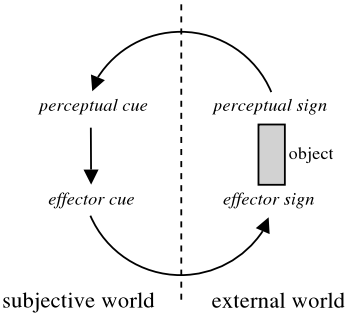


Fig. 1. A functional circle describes the abstract functional relationship between an organism and an object in its world. The perceptual sign of an object (its colour, shape, smell, or some more complex set of attributes) give rise to a perceptual cue which is defined as the subjective experience of the object in the organism's *Umwelt*. This leads to the creation of an effector cue which drives the animal to perform some action, thereby changing the organism's relationship to the object. A functional circle is an abstract description of a relationship, and it is hard to claim that a perceptual cue exists in any particular location, e.g. within a specific part of the nervous system.

First of all, the tick positions itself upon a suitable twig. If a mammal passes close by, the butyric acid emitted by the mammal provides a stimulus for the tick's smell receptor organ. This initiates a functional circle whose perceptual sign is the smell of the acid. A corresponding perceptual cue is produced, meaning that the mammal exists as an object in the tick's *Umwelt*. The resulting *effector cue* causes the tick to drop from the twig. The shock of the tick landing on the body of the mammal extinguishes the activation of the first functional circle. This means that the smell of the butyric acid no longer serves as a perceptual sign.

However, the shock of landing serves as the perceptual sign of a second functional circle. This initiates the behaviour in the tick of burrowing around in the mammal's fur until it encounters a patch of bare skin. The heat of the skin extinguishes the second functional circle, so the perceptual sign of the recent shock of the tick landing on the mammal no longer causes a perceptual cue and so the tick stops burrowing.

The third functional circle has the heat of the patch of skin serving as the perceptual sign. The heat produces a perceptual cue and so the patch of skin is experienced in the tick's *Umwelt*. The effector cue is produced and results in biting motions of the tick's mouth parts.

These actions can all be described as reflex behaviours with chemical or physical stimuli initiating fixed actions. Each behaviour has been selected by a process of natural selection to follow in the given order. We have used the model of von Uexküll's functional circles to explain them, which allow us to say that there is a perceptual cue associated with each circle.

If the organism is in a state suitable to allow the perceptual sign to serve as a stimulus, the perceptual sign causes the corresponding perceptual cue to be present within the constructed world of the organism. This is not to say that the perceptual sign *represents* some external object, or it is *presented* to the agent. As we shall see, the organism has chosen its own stimuli over its evolutionary history, and chooses whether or not to respond to it depending upon its current state. Both the stimulus, and the significance of the stimulus to the organism, is chosen by the organism. As a result, it may not be clear *what* the perceptual sign means to the organism.

A purely reactive organism relies on certain behaviours producing a predictable sequence of events in the environment that can be used as signs, e.g. an animal that hides in the dark depends upon the fact that the light intensity will decrease as it moves from the light thus extinguishing the stimulus. If the environment provided *all* the stimuli to cue the organism to produce its adapted behaviour, the organism would require only a simple and unsophisticated constructed world. However, a complex organism must engage in different sequences of actions at different times in an environment that provides the same potential stimuli. The organism must therefore choose which signs to respond to. A more complex organism is able to shape and vary how it perceives and hence responds to the world.

An organism with an *Umwelt* consisting of a small number of functional circles will perform only simple sequences of behaviour. This does not mean that the organism's physical environment is simple, but the environment *as perceived by the organism* is simple. The organism has selected over its evolutionary history the stimuli to which it should respond in order to engage with the world as reliably and efficiently as possible. Evolution acts to reduce the entropy of a constructed world despite of presence of environmental noise.

2.2 Self-extinguishing Functional Circles

Von Uexküll defined functional circles as always being self-extinguishing; that is, they produce behaviour in the organism that acts to remove the perceptual cue from the constructed world of the organism (figure 2). This can happen in two ways. The first way is that the functional circle causes the organism to alter its physical state to directly remove the stimulus, e.g. a photophobic organism will move away from a light, and thereby remove the stimulus that compels it to move. The second way is that one functional circle may *inhibit* the activation of another, e.g. the shock of the tick landing upon the mammal inhibits the first functional circle, even though the sign of the smell of butyric acid is still present.

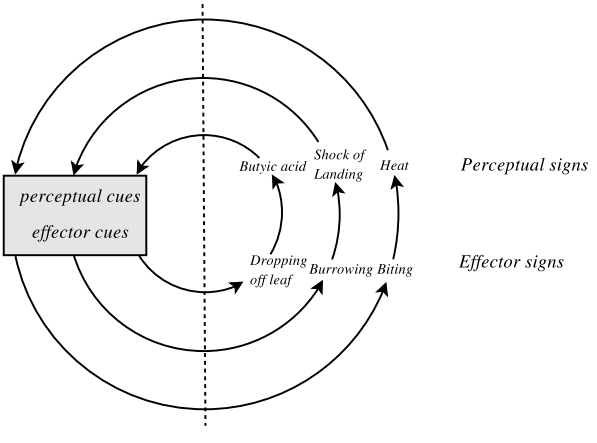


Fig. 2. The shaded region denotes that the functional circles negotiate among themselves as to which should be active. In the example of the behaviour of the tick, three functional circles negotiate. The perceptual cue of the first functional circle (i.e. the smell of the butyric acid), is inhibited and extinguished by the perceptual cue of the second functional circle (i.e. the shock of the tick landing upon the body of a mammal), and therefore no longer exists in the agents constructed world. This means that the butyric acid no longer acts as a stimulus, even though it may still be present, and the tick no longer perceives the smell of the acid. We can say that the two functional circles have negotiated to decide which should be active.

We can see that perceptual cues result in the creation of further perceptual cues, which act to extinguish the originator. The tick may either land on warm fur, or miss the mammal and hit the hard ground. The originating perceptual cue, the smell of the acid, invokes an action that results in one of two succeeding perceptual cues. Each extinguish the activation of the originator, and hence its associated behaviour or action.

2.3 An Organism Defines Its Own Perceptual Cues

A simple organism has evolved to respond to certain internal and external perceptual signs. The organism doesn't respond to a particular sign, e.g. the perception of heat, *because* it will get burnt, but because it has inherited a tendency to do so. The species to which an organism belongs has selected over evolutionary time the stimuli to which the organism should respond. The *meaning* of a perceptual sign may not be obvious to an outside observer, but it provides information that's meaningful to the organism that the organism uses to regulate or select its behaviour.

3 States of the Constructed World

3.1 It is Difficult to Classify States in Distributed Dynamic Controllers

It is necessary for an animat to co-ordinate its actions properly if its overall behaviour is to be purposeful. A simple behaviour is constructed from a number of discrete actions that occur one after another. The animat must know when to initiate which action, and at which point within the sequence, if the overall behaviour is to be successful. Traditional artificial intelligence techniques include the *finite state machine*. Using this technique, a distinct state and an action are associated together such that an animat in state A_s would always produce action A_a. Each state is linked to a limited number of other states. This reduces the possible sequences from action to action (i.e. state to state). The controller of such an animat is highly structured, and maps specific parts of the controller to specific local behaviours. This means modules that are responsible for producing individual behaviours can be programmed to react to specific stimuli and tested individually.

A continuous time recurrent neural network (CTRNN) controller is generally less formally structured and as a connectionist model has its activation states and any possible representations distributed throughout its structure. The state of each of its neurons can be altered by any other neuron it is connected to. This means it is not easy to claim a particular neuron or set of neurons of a CTRNN is responsible for a particular behaviour. CTRNN's use the idea of attractors, such that a CTRNN experiencing a particular attractor can be said to be in a particular state [8]. Desired actions are associated with different attractors. Attractors emerge from the dynamic activations of the CTRNN and are not

explicitly defined. It is therefore difficult to isolate a behaviour or for a single neuron to selectively generate a specific behaviour. It is left to the controller as a whole to co-ordinate different behaviours within itself. This makes the robot's behaviour difficult to analyse and understand. It also makes it difficult to reuse existing behaviours in different contexts, because the production of a behaviour may depend upon the state of a number of neurons, which in turn depend upon the state of further neurons, and so on.

Although it is an ongoing and current research issue, there is currently no clear general method to classify the instantaneous activations of the components of a CTRNN into states. It is a complex task to analyse even simple artificial neural systems [8]. The alternative proposed in this paper is to explain behaviour in terms of functional circles as emergent structures since functional circles are a simple and direct model of perceptions and behaviour.

3.2 Perceptual Cues can be Modelled as States

A perceptual cue associated with a behaviour in the organism results in the activation of one or more succeeding perceptual cues with varying probabilities. The succeeding cues extinguish their originator using one of the two methods described above (see 2.2). Using this model, the constructed world acts as a manager and decides which perceptual signs should invoke actions and which should be ignored.

We can classify a state of the constructed world as the instantaneous activations of its perceptual cues. The state changes as the perceptual cues run in sequence as part of the process of the embodied animat engaging with the world. This state doesn't represent information about the environment. It is information about the *world as perceived* by the animat. When we apply this model, we are able to treat the perceptual world as a series of states that lead to other states, regulated by, and influencing the behaviour of, the embodied animat acting in its environment. The meaning of the perceptual cues, and therefore the information that defines the state of the constructed world, has been chosen and defined by the organism over evolutionary time.

3.3 Implementing an *Umwelt* for an Animat

The CTRNN was originally advocated as part of the methodology of minimal cognition [10]. Minimal cognition is an attempt to study the simplest form of cognition using the assumption that to understand a complex phenomena, we start with something we can understand and gradually increase its complexity. However, there is a distinct difference between the first, simplest animals, and the animats we evolve for our evolutionary robotics experiments. Minimal cognition was inspired by those simple animals with co-ordinated nervous systems. However, it is likely that the behaviour of todays simple animals originated early in evolutionary time from simple, reactive, independent but co-operating reflexes (figure 4). The behaviour of simple animals is co-ordinated as they are comprised of sequences of actions that evolved to occur in a meaningful order. This doesn't

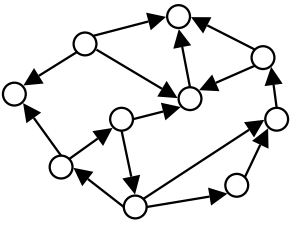


Fig. 3. It is difficult adding to an evolved robot's repertoire of behaviour because it requires integrating additional sensorimotor loops to an existing connectionist controller that defines preexisting behaviours. Sensorimotor loops in robots are partly realised as dynamic states along components of the controller. Some of these parts may be common to the action of several sensorimotor loops. A mutation to one of these common parts may be beneficial to one behaviour but detrimental to another; this means that sensorimotor loops from which each desired behaviour is composed can not evolve in isolation from each other, and this may constrain the evolution of both. Subsumption architecture [9] was developed in behaviour based robotics partly to solve the problem of scalability but no generally accepted methodology exists for evolutionary robotics.

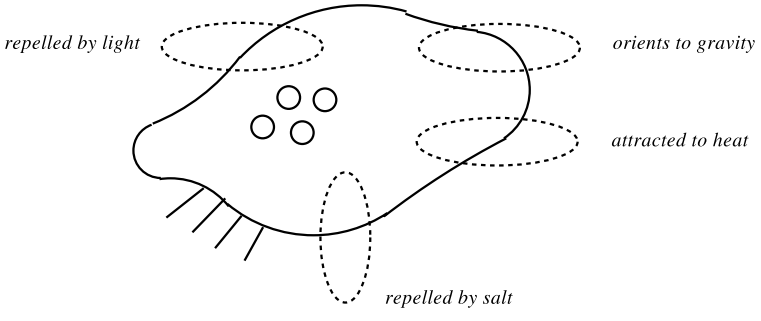


Fig. 4. Simple animals do not have a controller but operate through reflex reactions that are co-ordinated by the environment. Over evolutionary time, the organism may evolve more complex behaviour by negotiating and therefore selecting which reflexes to respond to.

mean that the environment is the controller of the organism. The agent within its environment reacts in a purposeful way to individual stimuli that it expects to occur in a certain order. The environment has to provide these stimuli in a given order for the agent to function properly.

Organisms without explicit controllers make up almost all living things, and yet they still manage to display adaptive behaviour. The functional circle model provides a union between the sequencing of actions and the actions themselves. Together with evolving perceptual cues, evolving negotiating functional circles allows agents to chose their own stimuli, and alter and reuse sequences of actions without altering the actions themselves. We can at some later point derive a co-ordinated control system developed from low-level reflexes.

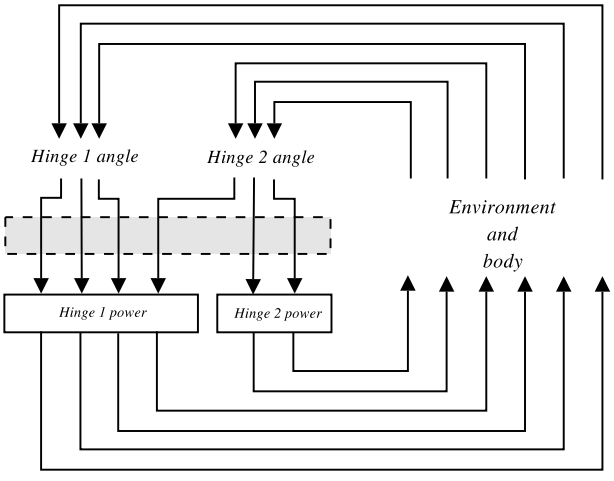


Fig. 5. Although functional circles can negotiate through the environment, evolved functional circles involving input sensors are likely to negotiate at the most mutable point, inside the controller. We should see the controller as mostly a negotiator of functional circles. We can view them as negotiating between perceptual cues.

It may not be possible to directly implement a model of functional circles (indeed every responding system can be described in terms of functional circles) but only to use mechanisms that encourage and highlight their essential properties. We can explain the behaviour of the emergent structures of CTRNN's in terms of functional circles. We often look at activated neurons that correspond with certain behaviours [8]. But we can alter our models to encourage the production of perceptual cues (figure 5).

The perceptual signs were clear and easily identifiable in the case of the tick. However, we do not need to know what perceptual signs the agent is reacting to if we chose a model whereby we can identify, by the nature of the model, its perceptual cues. We do not need to know or understand what the perceptual cue is actually reacting to. If we treat the controller as a negotiating part, and allow our perceptual cues to evolve separately, then we will be able to use this model.

The functional circle model can be applied to any the behaviour of any animat, including animats controlled by CTRNN's, even though the representation of their perceptual cues may be distributed across a network. All controllers can be described in functional circle terms because the *Umwelt* is an abstract model of behaviour and subjective experience, and how these affect the outside world.

4 Conclusion

Von Uexküll's original model of the *Umwelt* provides a model of an organism's constructed world. The constructed world is the environment as perceived by the

organism. This is an abstract model that can be used to describe the behaviour of any animat, regardless of how their controlling mechanisms are implemented. If we apply this model to describe the behaviour of the CTRNN controllers of our evolved animats, we benefit in two ways. The first is that by treating functional circles as emerged structures of CTRNN controllers and evolving the manner in which they negotiate together, we are evolving the relationships of an animat with its environment. The second is that we can define varying states for the constructed world of the animat. We can apply the tools associated with state machines while enjoying the benefits of a non-representational controller.

References

1. Kant, I.: A Critique of Pure Reason. 1930 edn. G. Bell and Sons, Ltd., London (1781) English Translation.
2. Brooks, R.: Intelligence without reason. In Myopoulos, J., Reiter, R., eds.: Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91), Sydney, Australia, Morgan Kaufmann publishers Inc.: San Mateo, CA (1991) 569–595
3. Ashby, W.: An Introduction to Cybernetics. John Wiley & Sons, Inc., New York (1956)
4. Strogatz, S.: Nonlinear Dynamics and Chaos. 2000 edn. Westview Press, Cambridge,MA (1994)
5. Avraham, H., Chechik, G., Ruppin, E.: Are there representations in embodied evolved agents? taking measures. In W.Banzhaf, T.Christaller, P.Dittrich, J.T.Kim, J.Ziegler, eds.: Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life (ECAL). Volume 2801 of Lecture Notes in Artificial Intelligence., Springer Verlag Berlin, Heidelberg (2003) 743–752
6. Klyubin, A., Polani, D., Nehaniv, C.: Tracking information flow through the environment: Simple cases of stigmergy. In Pollack, J., Bedau, M., Husbands, P., Ikegami, T., Watson, R., eds.: Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, Cambridge, Massachusetts, MIT Press (2004) 562–568
7. von Uexküll, J.: A stroll through the worlds of animals and men : A picture book of invisible worlds. In Schiller, C., ed.: Instinctive Behaviour: The development of a modern concept. 1957 edn. International Universities Press, Inc., New York (1934) English Translation.
8. Beer, R.: The dynamics of active categorical perception in an evolved model agent. Adaptive Behaviour **4** (2003) 209–243
9. Brooks, R.: Intelligence without representation. In Meyer, A.R., Guttag, J.V., Rivest, R.L., Szolovits, P., eds.: Research Directions in Computer Science: An MIT Perspective. MIT Press, Cambridge, MA (1991) 249–276
10. Beer, R.: The dynamics of adaptive behaviour: A research program. Robotics and Autonomous Systems **20** (1997) 765–776

Globular Universe and Autopoietic Automata: A Framework for Artificial Life*

Jiří Wiedermann

Institute of Computer Science,
Academy of Sciences of the Czech Republic,
Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic
jiri.wiedermann@cs.cas.cz

Abstract. We present two original computational models — globular universe and autopoietic automata — capturing the basic aspects of an evolution: a construction of self–reproducing automata by self–assembly and a transfer of algorithmically modified genetic information over generations. Within this framework we show implementation of autopoietic automata in a globular universe. Further, we characterize the computational power of lineages of autopoietic automata via interactive Turing machines and show an unbounded complexity growth of a computational power of automata during the evolution. Finally, we define the problem of sustainable evolution and show its undecidability.

1 Introduction

Some 50 years after von Neumann made public his result on self–reproducing automata it appears that this result should not be merely seen as convincing evidence of the existence of complex artificial self–reproducing structures. As McMullin pointed out [3], von Neumann's effort is to be also understood as the first step towards a construction of structures which exhibit an ability of evolutionary complexity growth *"from simpler types to increasingly complicated types",* as von Neumann put it. Unfortunately, von Neumann, in his work, did not tackled the issue of complexity growth of self–reproducing automata in more detail. In conclusion of his work [4] he merely indicated that these are the random mutations which should play the role of the respective evolutionary mechanism and included this problem into the list of problems which should be still investigated in the future (cf. [3] for a detailed discussion of that matter).

Obviously, von Neumann's intuition was right as we nowadays know from the theory of cellular and evolutionary biology. Nevertheless, a problem remains whether random mutations are the only mechanism for the artificial evolution, or whether there exists another mechanism resulting into more efficient evolution. A related question is whether the hypothetical "other" mechanism can become a

* This research was carried out within the institutional research plan AV0Z10300504 and partially supported by grant No. 1ET100300419 within the National Research Program "Information Society".

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 21–30, 2005.

subject of the evolution and how this can be achieved. Last but not least, is there an unbounded evolution, generating self–reproducing automata with increasingly complicated behavior, from a computational complexity viewpoint? What is the computational power of an unbounded evolutionary process? And can we decide whether a given self–reproducing machine will give rise to an endless evolution under given conditions? Some of these questions were discussed in [3], indicating possible ways to answer them.

In this paper we sketch formal answers to the above mentioned questions. For such a purpose we will present concepts and the results which emerge from the author's recent studies of the related problems. In their preliminary form the respective results were presented partly in a recent workshop [6] and partly as a technical report [7]. Contrary to these works the present paper is a survey paper stressing the main ideas behind the respective models and results rather than their formal description and proofs. This is so because a technical explanation would require more space than it is available in this contribution. The interested reader is invited to read the original sources.

In Part 2 we introduce an original model of globular universe which is the basis for construction of so–called self–reproducing globular automata. In Part 3 we informally define a simple abstract model of self–reproducing automata — so–called autopoietic automata — which obey self–reproducing abilities by their very definition. From their own genetic information these automata are able to compute new, algorithmically modified genetic information and pass it to their offsprings. In Part 4 we show that an arbitrary autopoietic automaton can be realized in a suitable globular universe. The reproduction of the resulting automaton proceeds, in a similar way, as the replication of a DNA strand in the living cells. Further on, we will deal exclusively with the autopoietic automata. In Part 5 we will characterize the computational power of the lineages of such automata by equalling it to the power of interactive Turing machines. We also mention the problem of a sustainable evolution and show its computational undecidability. Part 6 asks for the existence of an autopoietic automaton with an unliminted "self–improvement" property. Such an automaton initiates an evolution generating subsequently all autopoietic automata. This automaton controls its mutations is such a way that the genetic code "syntax" gets preserved; the evolution of the mutational mechanism itself guarantees the coverage of the entire evolutionary space. Section 7 is a closing part.

2 Globular Universe

The basic design idea of a globular universe comes from the following gedanken experiment with a classical cellular automaton. Imagine a standard two–dimensional cellular automaton and cut it by vertical and horizontal cuts into single cells each of which is seen as a single finite–state automaton. Doing so the cells will lose contacts with their neighbors. Let the cells freely fly around in the space, occasionally colliding one with each other much like as under the Brownian motion. Under a collision the cells come again into a passing contact

and on that occasion they perform a "computation" (a state transition) similarly as in the case when they were bound in a rigid grid of a cellular automaton. In our case, however, the colliding cells are not allowed only to change their states, they can also change properties of their contact domains. E.g., after the collision the originally neutral contact domains can become "sticky" and thus both cells will get bound together. Or the contact domains will become repulsive — the cells will be forced to move apart from each other. Thus, as a result we get a kind of a programmable matter, or a universe with programmable particles. The previous picture is not quite correct yet as far as the analogy with the Brownian motion is concerned. In our model we do not consider any kinetic aspects of cells moving in the space (like in lattice–gas automata); we are only interested in their final destinations, where they collide with an other object. Thus, to simplify the model further, we will assume that a cell in a proper state (i.e., having the required properties) "will come flying" to a place where we need it and at due time. Such an approach has great advantages over the probabilistic approach where we must care about probabilities by which the required phenomena occur. Our assumption resembles a nondeterministic choice: of all possible alternatives which can in principle occur we assume that the one that suits to our purposes will occur, indeed. Then, if adequately programmed a cell can be incorporated into an object at hand which in this way is being built by self–assembly. The last cosmetic change of the previous ideas is the transformation of square–shaped cells of the original cellular automaton into *globules*. They are all alike, of the same size and on their spherical surface on exactly (computationally) defined locations they have *contact domains* whose *attraction properties* are controlled by a *finite state mechanism* which is the same for all globules. As a result, the *globular universe* is created by an infinite multiset of globules with a fixed set of contact domains defined on their surface. The properties (i.e., the state plus the attraction abilities of contact domains) of globules at interaction times are controlled by so–called *interaction function* (or relation) which says how the states and attraction properties of two interacting globules will change after the interaction (for more details see the original paper [6]). A somewhat similar experimental framework using a finite state mechanism to model contact properties of polyhedral particles moving in a fluid in a simulated physical setting has been described in [2].

It is clear that our model of globular universe is a generalization of both classical cellular automata and contemporary models of self–assembly (cf. [1]) as used in computer science.

3 Autopoietic Automata

Our next goal will be a construction of a so–called self–reproducing globular automaton. Its construction will be based on the principles of self–reproduction as "discovered" by von Neumann [4]: the same "program" will be used both for controlling automaton's own computational behavior and as a template for the production of an other program which will later control the offspring of

the self–reproducing automaton at hand. In a globular universe we cannot take advantage of the fixed grid structure underlying the classical cellular automaton as it was the case with the von Neumann's design. Namely, we cannot construct a copy of the original machine at a priori computed "free" grid locations in a sufficient distance from the parental machine. Instead, we will make use of the self–assembly properties of the elements of a globular universe. Prior to immerging into a design of a self–reproducing globular automaton we describe a more abstract object, a so–called autopoietic automaton[1] whose formal definition can be found in [7]. This (mathematical) object will serve as a kind of abstract specification of a self–reproducing globular automaton. An implementation of an autopoietic automaton in a suitably designed globular universe we give rise to the required self–reproducing globular automaton.

Autopoietic automata are nondeterministic finite state machines capturing the elementary information processing, reproducing and evolving abilities of living cells. Technically, an autopoietic automaton is a nondeterministic transducer (a Mealy automaton) computing, in addition to the standard translation also the transition relation of its offspring. The design of an autopoietic automaton supports working in two modes. Both modes are controlled by a transition relation. The first of them is a standard *transducer mode* in which external input information is read through an input port. In this phase the results of a computation (if any) are sent to the output port. The second mode is a *reproducing mode* in which no external information is taken into account. Instead, the representation of a transition relation itself is used as a kind of the internal input. For this purpose the representation of automaton's own transition relation is available to an autopoietic automaton on a special, so–called *program tape*. It is a two–way read–only tape. The results of the computational steps in the reproducing mode are written on a special one–way write–only *output tape*. Of course, both tapes mentioned before are finite.

In general, the *transition relation* of an autopoietic automaton is a finite subset of the cartesian product $\Sigma \times Q \times \Sigma \times Q \times D$, where Σ is an ordered set of input and output symbols, Q is the ordered set of states and D is the ordered set of move directions of the head on the program tape. Both sets Σ and Q can be infinite, whereas $D = \{d_1, d_2, d_3, d_4\}$. On the program tape the elements of these sets are unary encoded, i.e., an element $\sigma_i \in \Sigma$, $q_i \in Q$, or $d_i \in D$ is encoded as 0^i (i.e., as the string consisting of i zeros). It follows that a tuple $(\sigma_i, q_j, \sigma_k, q_m, d_n) \in \Sigma \times Q \times \Sigma \times Q \times D$ is represented on the tape as $10^i10^j10^k10^m10^n1$. Tuple $(\sigma_i, q_j, \sigma_k, q_m, d_n)$ is called an *instruction* of the transition relation; a representation of an instruction on the tape is called a *segment.* The semantics of an instruction is as follows: *"reading σ_i in state q_j*

[1] The name of autopoietic automata has been chosen both to honor the Chilean biologists Varela and Maturana who coined the term of autopoiesis and also to distinguish these automata by the name from the notoriously known classical notion of self–reproducing automata which are a kind of cellular automata. The autopoietic automata are definitely not meant to model autopoiesis in the sense of Maturana and Varela.

the automaton outputs σ_k, enters state q_m and shifts its head in direction d_n," where the value $n = 1$ means the shift by one cell to the left, $n = 2$ to the right, $n = 3$ means no shift and $n = 4$ means "undefined". The segments are written on the program tape one after the other.

Formally, the mode of activities of an autopoietic automaton are derived from the type of state that the automaton is in at that time. For such a purpose the states of an automaton are split into two disjoint sets: a translating and a reproducing set. To distinguish the types of individual states in their tape representation also syntactically we will make use of the last component in the five–tuple representing a segment. When, after entering state q_m, the automaton should work in a translating mode (i.e, when q_m is a translating state), the component in the respective instruction will take value d_4 and in the respective segment value 0^4. Otherwise, to indicate the reproducing states, this component will take values d_1, d_2, or d_3 denoting the move direction for the program tape head. An autopoietic automaton will start its activity in an initial translating state and while remaining in this type of states the automaton continues working in the translation mode. The respective instructions are characterized by value d_4 in their last component signalling the absence of head moves. After the first entering a reproducing state the automaton must stay in the reproducing mode. The reproducing mode terminates by entering the final reproducing state. At that moment the automaton splits into two automata by definition. The first of the two will "inherit" the program tape of the parental automaton as its own program tape (denoted as Program 1 in Fig.1), whereas the second one will use, in place of its program tape, the output tape of the parental automaton (denoted as Program 2). Then both new automata start their activities with empty output tapes. Each new automaton is seen as an offspring of the original automaton. Clearly, one of the offsprings will always be identical to its parent, but the other offspring can be different from its parent, indeed. Note that it has been the admittance of infinite symbol and state sets allowing an autopoietic automaton's offspring to work with a larger set of symbols or states than its parent could. By this we have opened a possibility of an evolution leading from simpler to more complicated automata.

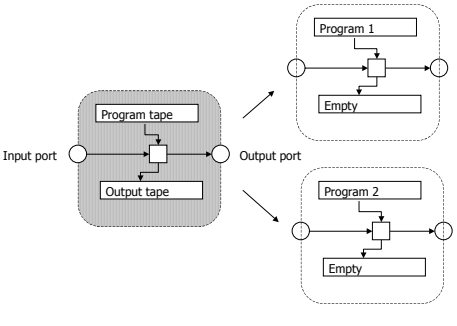


Fig. 1. Autopoietic automaton reproducing by fission

4 Implementing Autopoietic Automata in a Globular Universe

Now we design a specific globular universe and show the implementation of autopoietic automata in it. It is obvious that autopoietic automata cannot exist in a universe which is "too simple". For instance, a universe with only single–state globules does not enable any interaction of globules that would result in their state changes. Similarly, no globular complexes can be built in a universe with only neutral globules (i.e. with those unable entering attraction states). In the sequel a universe in which we will implement an autopoietic automaton will not be explicitly described. Rather, the required properties of the globular universe will be implied by the construction of the self–reproducing globular automaton and it will be clear that such a universe does exist, indeed.

Theorem 1. *There is a nondeterministic globular universe in which, for any autopoietic automaton, there exists its implementation in the form of a self–reproducing globular automaton.*

Sketch of the proof: Let $\mathcal{A}$ be an autopoietic automaton and consider its program tape with the segments of the transition relation of $\mathcal{A}$ written on it. In our universe we will represent this tape as a string of globules. The length of this string equals that of the tape. For simplicity we will first assume that globules in our universe have enough states for directly representing the symbols of a finite subset $S \subset \Sigma$ and $R \subset Q$ really used by the automaton. This means, we assume that there is a one–to–one correspondence between the set of states of globules and S or R, respectively. Moreover, assume that some extra states still remain free. These states will be used to hold "auxiliary variables" in our construction. Thus, in our string each globule will be in a state which uniquely corresponds to the symbol encoded on the automaton's program tape. The globules are designed so that they have four equidistant contact domains — poles — around their equator. On each globule one pair of opposite poles is in a "sticky" state forcing globules to form a sequence. Moreover, we can assume that the first and the last globule will also stick together giving rise to a so–called basic ring representing the transition relation of $\mathcal{A}$. This ring will form the basis of the globular automaton $\mathcal{G}$ implementing $\mathcal{A}$.

Now it is time to describe the input "mechanism" by which $\mathcal{G}$ reads its inputs. We will simply assume that $\mathcal{G}$ "reads" its input by its entire "body". I.e., we will assume that there will come a "wave" of the input globules which will attach themselves (because they are programmed so) to the globules' poles located on the same side of the basic ring. We can assume the existence of such a wave thanks to the properties of the nondeterministic universe. Afterwards, $\mathcal{G}$ starts to work in the translation mode as an interpreter of the code which is represented in the basic ring. In order to work in this way there is an additional ring attached to the basic ring where additional globules representing necessary auxiliary "variables" are kept. The globules in this ring serve e.g. as "markers" denoting the current state or the segment with an instruction to be performed,

the program head location, by their interaction "circular" signals are sent around the rings, etc. The resulting globular automaton takes a form of a double ring; its globules mirror the actions of a similar classical cellular automaton. The only difference is that $\mathcal{G}$ is "made of" globules instead of cells, but the neighboring spacial relations among the globules and cells are the same. The details of the construction are given in [6]. The output from $\mathcal{G}$ is done in an analogous way to the input, i.e., the original input globules are "transformed" (via a change of their states) into required output globules and are "released" into the environment (via a change of their attraction properties). In this way the actions of $\mathcal{G}$ proceed until $\mathcal{G}$ enters the first reproducing state.

In the reproducing mode $\mathcal{G}$ slavishly interprets the instructions from its program tape similarly as before. The difference is that now the input is read directly from the globules of the basic ring (for such a purpose the head position must be represented in the auxiliary ring) and the output globules (to which the incoming globules are transformed) are not all released into the environment but those at proper places are subsequently "glued" together to form yet another ring which grows in this manner. In parallel to this ring an auxiliary ring much as the one attached to the basic ring is built. The emerging output and auxiliary rings also form a double ring which touches the original rings at the place where the reading head was initially located. Once $\mathcal{G}$ enters the final reproducing state, the newly generated double ring detaches itself from the original double ring and each complex starts to exist as an independent self–reproducing globular automaton.

The implementation of $\mathcal{A}$ we have just described works in a universe with globules having a sufficient number of states. In a universe with less states we have to work with globules corresponding to the unary coding of states and symbols as required by the definition of autopoietic automata. Thus the simulation process gets more complicated; nevertheless, its main features will remain the same. □

Even from the previous sketch one can see that in order to realize a self–reproducing globular automaton a nondeterministic universe with a certain minimal number of states is needed. The upper bound on this number could be inferred from a more detailed description of our construction. On the other hand, it is also clear that in universes with too small a number of globular states it might not be principally possible to build a self–reproducing automaton.

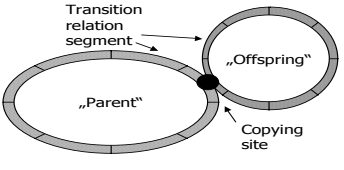


Fig. 2. Self–reproduction of a globular automaton

5 The Computational Power of Autopoietic Automata

In order to get an idea of the computational power of self–reproducing globular automata we will study the power of autopoietic automata. The next theorem will characterize the computational power of the so–called lineages of autopoietic automata with the help of interactive Turing machines. A *lineage of autopoietic automata* is an infinite sequence of autopoietic automata in which each member has exactly one immediate successor which is an immediate offspring of that member. A lineage corresponds to a single path, starting in the root, in a "genealogical" tree consisting of all possible offsprings of a given autopoietic automaton which is located in the root. In this tree, parents are linked to their immediate offsprings. A tree will emerge due to the fact that a single autopoietic automaton can give rise to several offsprings. An *interactive Turing machine* (ITM) is a Turing machine reading a potentially infinite sequence of its inputs via an input port and sending its outputs to the output port [5]. Both in the case of autopoietic automata and ITMs we allow so–called empty inputs that correspond to a situation when no symbol from Σ appears at some port at that time. We say that an ITM *simulates a given autopoietic automaton* (or vice versa) if and only if both devices compute the same translation (mapping) from the input symbol sequence to the output symbol sequence, with empty symbols deleted from both sequences.

Theorem 2. *The computational power of a lineage of (nondeterministic) autopoietic automata equals to that of a nondeterministic interactive Turing machine.*

Sketch of the proof: The proof of the left–to–right implication is relatively simple. The simulation of a given member of a lineage is carried out by the universal ITM which, on its first working tape, has a representation of automaton's program tape and interprets the instructions from this tape. In the translation mode the machine reads its inputs from the input port and sends the outputs to its output port. In the reproducing mode the machine reads its first tape and writes the output to the second working tape. After reaching the automaton's final reproducing state the machine changes the role of its tapes, empties the second tape and the simulation of the next member of a lineage can resume.

The reverse simulation is more complicated. The idea is to see the ITM's computations performed in the space of size i as those of the finite state automaton A_i, for $i = 1, 2, \ldots$. The automata A_i are realized by corresponding autopoietic automata. What must be designed is the reproducing instructions for autopoietic automata which, as one can see, are the same for all automata. Their task is to "compute" the transition relation for A_{i+1} given the transition relation of A_i. Then, along with the growing space complexity of the Turing machine increasingly bigger autopoietic automata are generated giving the members of the required lineage we are after. For more details see the original report [7]. □

It is obvious that the halting problem for an ITM is undecidable. Thus, it is undecidable whether, given an infinite input sequence and an ITM, the

machine will ever halt on that input. A similar question for a given autopoietic automaton and a given infinite input sequence would be the following *problem of a sustainable evolution*: is it decidable whether the automaton will generate an infinite lineage of its offsprings on that input? Referring to the previous theorem the following result holds:

Corollary 1. *The problem of a sustainable evolution is undecidable.*

6 Unbounded Evolutionary Complexity Growth

The previous corollary shows that for a given input sequence we cannot in general decide whether an autopoietic automaton will generate an infinite lineage. It is trivial to see that the situation changes dramatically if we submit to the automata suitable inputs that will cause their entering final reproducing states. To see this it is enough to consider an automaton which replicates on some input and to submit the same input to that offspring which equals its parent. But now we present a much less obvious result — we show an unbounded complexity growth of automata during an evolution by constructing an automaton which in a suitable nondeterministic universe generates all possible autopoietic automata.

Theorem 3. *There exists a nondeterministic autopoietic automaton which, in a suitable nondeterministic universe, generates all existing autopoietic automata.*

Sketch of the proof: We design an automaton whose code contains only the reproducing instructions. These instructions read the segments from the automaton's program tape, modify them and rewrite them onto the output tape. The modifications do not concern the syntax of the segments, i.e., the separators (symbols 1) between the sectors as well as the number of sectors remain unchanged. Then the modifications are threefold:

- A change within a sector: when copying a sector the automaton adds or omits one zero; it follows that the successor automaton will work with other symbols or states than its parent;
- Adding one segment whose contents is nondeterministically generated; this will potentially give rise to a "more complex" automaton;
- Omitting a segment (the automata get "simplified").

It is clear that the initial automaton will subsequently generate autopoietic automata with all possible transition relations. Those of these automata which could in principle reach the final reproducing state on some input will indeed get such an input and hence will self–reproduce. The other automata will not reproduce. Again, for the details see the original report [7]. □

It is important to realize that the construction just described enables an evolution of the mechanism which is responsible for the algorithmic modification of the transition relation. Thus, what we got is a mechanism for the evolution's evolution. Putting it differently, in our latest autopoietic automaton the evolution

is not guided by a fixed set of rules; rather, these rules themselves are subjects of an evolution. This is a property not possessed by the automata constructed in the proof of Theorem 2. Last but not least, note that for covering the whole evolutionary space of autopoietic automata it was of fundamental importance that both components of an autopoietic automaton's control — the translating and the reproducing one — were a subject of an evolution.

7 Conclusion

In the paper we surveyed several fundamental results concerning the self–reproducing automata. All these results were based on a new formal framework enabling computational modelling and mathematical analyzing of the respective phenomena. The results show the viability of the approach, bring new insights into the nature and power of evolutionary processes and thus are of interest both from the artificial life as well as from the computational complexity theory point of view.

References

1. Adleman, L.: Toward a mathematical theory of self-assembly. Tech. Rep. 00-722, Dept. of Computer Science, University of Southern California, 2000.
2. Dorin, A.: Physically Based, Self-Organizing Cellular Automata, in Multi–Agent Systems — Theories, Languages and Applications, Zhang C., Lukose D. (eds), Lecture Notes in Artificial Intelligence 1544, Springer-Verlag, 1998, pp. 74-87.
3. McMullin, B.: John von Neumann and the Evolutionary Growth of Complexity: Looking Backwards, Looking Forwards... Artificial Life, Vol 6. Issue 4, Fall 2000, pp. 347-361
4. von Neumann, J.: Theory of Selfreproducing Automata. A. Burks (Ed.), University of Illinois Press, Urbana and London, 1966
5. van Leeuwen, J., Wiedermann, J.: The Turing machine paradigm in contemporary computing, in: B. Enquist and W. Schmidt (Eds), *Mathematics Unlimited – 2001 and Beyond*, Springer-Verlag, Berlin, 2001, pp. 1139-1155.
6. Wiedermann, J.: Self–reproducing self-assembling evolutionary automata. Proc. of the Workshop on Tilings and Cellular Automata, WTCA'04, CDMTCS, University of Auckland, Dec. 2004.
http://www.cs.auckland.ac.nz/CDMTCS//researchreports/253maurice.pdf
7. Wiedermann, J.: Autopoietic automata. Research Report V-929, Institute of Computer Science AS CR, Prague, February 2005, cf. http:=//www.cs.cas.cz

May Embodiment Cause Hyper-Computation?

Jozef Kelemen

Institute of Computer Science, Silesian University,
746 01 Opava, Czech Republic and
VSM School of Management,
851 04 Bratislava, Slovak Republic
kelemen@fpf.slu.cz

Abstract. The contribution provides an example of how a formal model of some life-like functions – the so called eco-grammar (EG) system – provides a framework in which it is formally provable that the computational power of the model – under some very natural circumstances derived from the specificities of living systems, esp. from their embodiment – may overcome the computational limits of traditional computing models, perhaps also the computational power of the universal Turing machine.

1 Introduction

Starting from the original inspirations and ideas of the *Artificial Life* (AL) we are *concerned with tuning the behaviors of low-level machines so that the behavior that emerges at the global level is* essentially *the same as some behavior exhibited by natural living systems* [5, p. 4]. During the history of AL many of experiments and theoretical frameworks have been proposed in order to discover or to describe such type of emergence of the *life-like behaviors* from the *machine-like behaviors*. In [6] several suitable examples of the proposed approaches are listed as well as a hint – in the form of a specific *test of emergence* – how to valuate them in order of answer the question whether or not the life-like behaviors emerge from the machine-like ones. The test of emergence requires some principal *surprise* when we compare the behavior of simple machines forming the whole systems, and the behavior of the whole system. In fact, esp. this *principal* surprise proves that the behavior of the whole really emerges from the behaviors of its parts.

At least a part of AL research is in experimental level concerned to computers and in the theoretical level to tools and methods provided by *theoretical computer science*. From the point of view of theoretical computer science *abstract machines* – and formal models theoretically equivalent with them in formally well-defined sense, e.g *formal grammars*; cf. [4] – cannot do more as produce (accept, recognize) behaviors which can be produced (accepted, recognized) by the *universal Turing machine*. This is the core statement of the so called *Church-Turing Thesis*; cf. [8]. All of the arguments for the thesis, as Sieg pointed out, take for granted that the computations are being carried out by human beings, and that the computations are taken mechanical. Ideas about incorporation of certain variant of "parallelism" and of

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 31 – 36, 2005.

overcoming in certain way the limits of the "mechanical" seems to be helpful for speculations on some new type of computing devices, might be inspired by some approaches of AL.

Going back to the *test of emergence*, it will be *principally surprising*, at least from the point of view of theoretical computer science, if in a well-formalized theoretical model of computations or computing machines it happens to prove that processes correctly described in the frame of this model go beyond the computational power of the universal Turing machine, if so – following the terminology proposed in [1] – the models are *super-recursive*, and the processes executed by them are *hyper-computations*; for some examples see [1].

In this contribution we will sketch a formal framework of eco-grammar systems proposed in [2] inspired by some features of living systems, and developed for studying the interplays of (formal) descriptions of behaviors of simple machines, and the results of these interplays. Then we will recall a result from [9] in order to illustrate the surprise: Eco-grammar systems, when we include a very natural feature of living systems – their *embodiment* – into them in certain way, are super-recursive models, and they are able to execute hyper-computations. So, by the example we will try to provide a formal proof supporting the intuitive idea that behaviors of living systems may have in certain situations the character of super-computations, so that they may go in certain situations beyond the behaviors of individually recursive abstract machines which, working together, may form super-recursive computing devices.

2 Eco-grammar Systems

According [2], an *eco-grammar* (*EG*) *system* Σ consists, roughly speaking, of

- a finite alphabet V,
- a fixed number (say n) of agents, and evolving according set of rules $P_1, P_2, ..., P_n$ applied in a parallel way as it is usual in the theory of L-systems [7], and of
- an environment of the form of a finite string over the set V (the states of the environment are described by strings of symbols w_E, the initial state of the environment is described by the string w_0).

The rules of agents depend, in general, on the state (on the just existing form of the string) of the environment. The agents act on a shared string (the state of the environment) by sets of their own sequential rewriting rules $R_1, R_2, ..., R_n$ as it is usual in the traditional formal language theory; cf. [4].

The evolution of the state of the environment, caused by agents, proceeds in the following way:

By the definition, a string w_i is rewritten into the string w_j ($w_i, w_j \in V^+$) according the mode of rewriting t (in short $w_i \Rightarrow^t w_j$) iff all agents which are competent to rewrite symbols if the corresponding EG system Σ works in the rewriting mode t are also able to execute the rewritin. The simplest case of the mode t is the one according which all of agents which *can* rewrite a symbol *must* execute the rewriting.

After the execution of a step of rewriting process by agents it follows the parallel rewriting according the rules from the set P_E.

The environment itself evolves thanks to and according the set P_E of rewriting rules applied in parallel as usual in L systems.[1]

The model is schematically depicted in the figure Fig. 1. We note, that the role nor the definition of the functions φ and ψ appearing in the formal model of eco-grammar systems as defined in [2] as well as in Fig. 1 will not be discussed in the following, so we will pay no attention to their definitions as well as specifications of their roles.

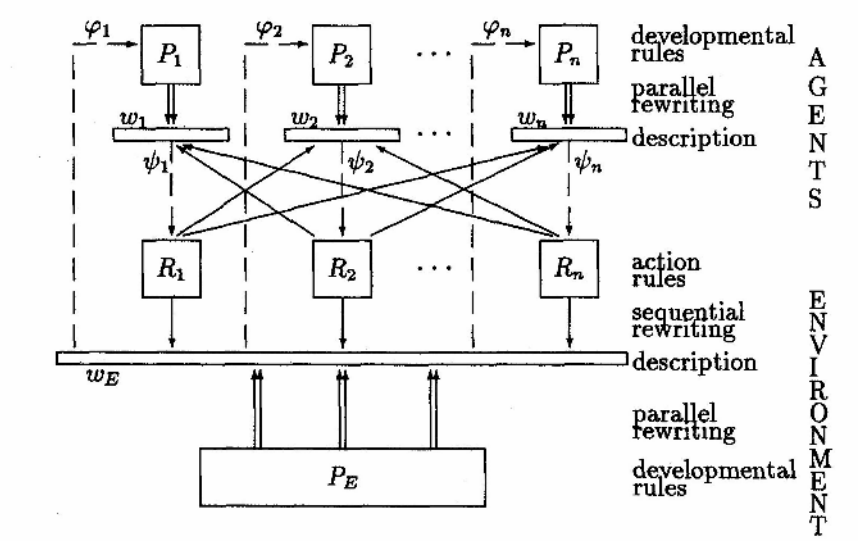


Fig. 1. A schematic view of a traditional EG system [2]

The evolution rules of the environment are independent on agents' states and of the state of the environment itself. The agents' actions have priority over the evolution rules of the environment. In a given time unit, exactly those symbols of the environment that are not affected by the action of any agent are rewritten.

In the EG-systems we assume the existence of the so called *universal clock* that marks time units, the same for all agents and for the environment, and according to which the evolution of the agents and of the environment is considered.

3 Teams in Eco-grammar Systems

In [3] a special variant of EG systems have been proposed in which agents are grouped into subsets of the set of all agents – into the so-called *teams* – with fixed number of agents. A more adequate from our perspective model has been proposed in [9] where a variant of EG systems without internal states of agents is studied, the fixed number of members proposed in [3] is replaced by a dynamically changing number of agents in teams.

[1] So, the triplet (V, P_E, w_E) is (and works as) an L system.

As the mechanism of reconfiguration in the proposal by D. Wätjen, a function f is defined on the set N of integers with values in the set $\{0, 1, 2, \ldots n\}$ (where n is the number of agents in the corresponding EG-system) in order to define the number of agents in teams: For the i-th step of the work of the given EG-system, the function f relates a number $f(i) \in \{0, 1, 2, \ldots n\}$. The subset of the set of all agents of thus EG-system of the cardinality $f(i)$ is then selected for executing the next derivation step of the EG system working with Wätjen-type teams.

Wätjen in [9] proved, roughly speaking, that there exist EG systems such that if f is (in the traditional sense) non-recursive function, then the corresponding EG system generates a non-recursive (in fact it may be also a super-recursive) language.

The language is defined in the case of Wätjen's type EG systems, say a system Σ, using in each derivation step only agents from the corresponding subset of the cardinality $f(i)$ of the set of agents of Σ, so:

$$L(\Sigma, f) = \{v \mid w_0 \Rightarrow^{f(1)} w_1 \Rightarrow^{f(2)} \ldots w_r \Rightarrow^{f(r)} = v,\ r \in N,\ w_0 \ldots w_r \in V^*\}$$

Consider the following example of a Wätjen's type EG-system which uses a function f [9]:

$\Sigma = (V, P_E, R_1, R_2, \ldots, R_n, w_E)$
where
$V = \{a, b, b_1, b_2, \ldots, b_n\}$,
$P_E = \{a \to a^2,\ b \to b^2\} \cup \{b_i \to b^2 \mid i = 1, 2, \ldots, n\}$,
$R_i = \{b \to bb_i\},\ 1 \le i \le n$,
$w_0 = a^2 b^{2n+3m},\ m, n \in N$.

This EG-system generates the following language:

$$L(\Sigma, f) = \left\{a^2 b^{2n+3m}\right\} \cup \bigcup_{k \in N} \left(\bigcup_{\substack{1 \le i_1, \ldots, i_{f(k)} \le n \\ (i_j \ne i_{j'},\ j \ne j',\ 1 \le j, j' \le f(k))}} \left\{a^{2^{k+1}}\right\} perm\left(bb_{i_1}, \ldots, bb_{i_{f(k)}}, \underbrace{b^2, \ldots\ldots\ldots\ldots, b^2}_{2^{k-1}(2n+3m)-f(k)\ times} \right) \right)$$

This language is *recursive*, if the function f is *recursive*. Then Wätjen supposes that in the case of a *non-recursive* f the language $L(\Sigma, f)$ remains *recursive*, and demonstrates a contradiction which follows from this: If $L(\Sigma, f)$ is recursive, then all the words of it can be effectively listed in some order. Now, chose an arbitrary $k \in N$. Then there exists a word w_k in $L(\Sigma, f)$ listed after a finite number of steps, so, we can compute the value $f(k)$ for it. From this fact follows that f is computable. This is, however, a contradiction! Because of this, the language $L(\Sigma, f)$ is *non-recursive* if f is *non-recursive*.

4 Concluding Notes on Embodiment

Incorporation of physical bodies of agents into the pure formal framework proposed for study of emergent computational properties of multi-agent systems consisting of

embodied individually autonomous agents is in general a hard problem for the formal models proposed in AL, esp. because there are no adequate formal tools at hand for involving the agent bodies into computational frameworks. EG systems in the form proposed in [2] incorporate several life-like features, but not the embodiment of agents.

However, in the context of EG systems with agents organized into *teams* as proposed in [3] and in [9] we may imagine and suppose – in a very intuitive level – that the physical bodies of agents are the very things we want for prohibition of agents activities in certain extent during the rewriting process. In the case of the fixed number of agents in teams [3] this kind of modeling the embodiment have no principal influence to the computational power of the EG systems. But in the case of changing number of agents in teams – as proved in [9] and as we sketched in the previous Section – the computational power of the EG systems basically depends on the computational properties of functions which define the number of active agents in teams. Non-recursive functions appearing in this model cause the possibility of generation of non-recursive languages – behaviors – using the corresponding EG systems. The non-recursivness of the functions f in the model can be interpreted as certain level of incorporation of the randomness into the models.

We conjecture that if this randomness incorporated into the EG systems will be at the level expressible only through hyper-computable functions f, then the corresponding EG systems which will use such type of functions will be – using the terminology proposed in [1] – *super-recursive*, so that they will be able to perform *hyper-computations*.

In any case, the interpretation of the result from [9] concerning the computational power of EG systems proves that embodiment of agents significantly influences the computational power of communities of agents in comparison with the individual computational power of agents forming these communities and participating on their activities. In certain sense the difference between the behavior of the agents and of the whole communities formed by them may be comprehended as a surprise required in the test of emergence formulated in [6], and the non-recursivity (or the hypothesized above super-recursivity) of the societies of agents as an emergent effect of agents embodiment.

Acknowledgment. Author's research on the subject is sponsored by the grant No. 201/04/0528 of Grant Agency of the Czech Republic, and by Gratex International Corp., Bratislava, Slovak Republic.

References

[1] Burgin, M., Klinger, A. (guest editors): Super-recursive algorithms and hyper-computation (special issue). *Theoretical Computer Science* **317** (2004) 1-267

[2] Csuhaj-Varju, E., Kelemen, J., Kelemenova, A., Paun, G.: Eco-grammar systems – a grammatical framework for lifelike interactions. *Artificial Life* **3** (1997) 1-28

[3] Csuhaj-Varju, E., Kelemenova, A.: Team behavior in eco-grammar systems. *Theoretical Computer Science* **209** (1998) 213-224
[4] Hopcroft, J. E., Ullman, J. D.: *Formal Languages and their Relation to Automata*. Addison-Wesley, Reading, Mass., 1969
[5] Langton, C. G.: Artificial life. In: *Artificial Life* (Langton, C. G., Ed.) Addison-Wesley, Redwood City, Cal., 1989, pp. 1-47
[6] Ronald, E. M. A., Sipper, M., Capcarrére, M. S.: Design, observation, surprise! A test of emergence. *Artificial Life* **5** (1999) 225-239
[7] Rozenberg, G, Salomaa, A.: *The Mathematical Theory of L-systems*. Academic Press, New York, 1980
[8] Sieg, W.: The Church-Turing thesis. In: *The MIT Encyclopedia of the Cognitive Sciences*. The MIT Press, Cambridge, Mass., 1999, pp. 116-117
[9] Wätjen, D.: Function-dependent teams in eco-grammar systems. *Theoretical Computer Science* **306** (2003) 39-53

Perception as a Dynamical Sensori-Motor Attraction Basin

M. Maillard[1], O. Gapenne[2], L. Hafemeister[1], and P. Gaussier[1]

[1] Neuro-cybernetic team, Image and signal processing Lab.,
ETIS, UMR CNRS 8051, Cergy Pontoise University,
ENSEA, 6 av. du Ponceau, 95014 Cergy, France
maillard@ensea.fr,
http://www.etis.ensea.fr/~neurocyber
[2] COSTECH Université de Techonologies de
Compiègne groupe suppléances perceptives
olivier.gapenne@utc.fr,
http://www.utc.fr/gsp/accueil.html

Abstract. In this paper, we propose a formal definition of the perception as a behavioral dynamical attraction basin. The perception is built from the integration of the sensori-motor flow. Psychological considerations and robotic experiments on an embodied "intelligent" system are provided to show how this definition can satisfy both psychologist and robotician point of view.

1 Introduction

The classical conception of perception refers to a parallel and passive computation of an input flow of information. In this frame a cognitive system is considered only as a computational system receiving inputs (namely the "sensations") to identify objects or events and producing output representations useful for reasoning and leading to appropriate actions. In this paper we investigate an alternative approach where perception is linked to dynamical laws between actions and sensations [22,14]. But the lack of a formalism leaves the community without tools to analyse this kind of concept. Therefore, taking avantage of a generic formalism developped to analyse cognitive systems [5], and refeering to psychological and neurobiological assumptions, we propose to define the perception as a dynamical sensori-motor attraction basin. Specifically, the formal description of a sensori-motor system leads us to define the perception in regard of the dynamics of the sensori-motor system. Moreover, we will show that a sensori-motor learning in a competitive neural network allows to approximate such an attraction basin. The immediate benefit is to be able to explore the concept by observing the resulting perception in robotic experiments and to confront it to psychological experiments.

2 Background Considerations

We briefly wish to recall that our formalism of perception inherits from a recent philosophical and scientific tradition. Indeed, during the 19th century, and more

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 37–46, 2005.

definitively during the 20th century, a whole philosophical work [9,17] states that the conscious experience and knowledge are the fact of a construction (third person) or constitution (first person). This poses that to know and perceive in an organized way is not given and supposes developmental and learning processes. Moreover, this epistemology of the construction or constitution which renews the statute of objectivity deeply affirms the role of the action and its effects as a condition of possibility and constraint [16]. And very radically, it poses that the lived experience of the world and oneself (and their relation) are defined by the properties of the actions system available to the agent which organizes the organism-environment relations. This led to the idea that the causality of the experience cannot be reduced to a strictly active or passive internal construction. In the scientific field, this approach had some eminent representatives so much in the life sciences [12,4] that in the social sciences [15,21,13]. There is obviously no question of developing the whole of this work here. We simply propose to specify the importance of the action and more precisely of some to know about this same action so that the process of perceptive genesis can take place. The perceptual knowledge associated with the action is classically described like concerning the proprioception. The latter indicates a system of coupling which intervenes in the perception of the movement (kinesthesis) and body positioning (statesthesis). Like any system of coupling, it implies a whole of elements which are not limited to particular sensors (neuro-muscular spindles, neuro-tendinous bodies or articular sensors) and their mode of transduction. This system implies a nervous network (cortex included in particular sensory-motor, premotor, left parietal, cingular bilateral cortex and supplementary motor area), effector (the muscles) and includes the environmental constraints (gravity, friction). Without going into the details of the operation of this coupling, it is possible to mention some significant points:

- Each movement is associated to a specific reafferent sensory flow which can be defined like a true signature.
- This signature is currently described in the shape of a vector which includes acceleration, speed, direction and duration of each movement.
- Formally, the relation between movement and sensory reafferences is bijective what guarantees a great stability of the invariants that this coupling authorizes.
- This system is continuously activated but some forms of adaptation at the time of prolonged immobility can be observed.

The proprioceptive coupling thus allows the constitution of reliable invariants relative to the body by convening the body itself. This sensitivity of the proprioceptive system to the only directed deformations of the body confers a particular statute to him to the glance of two other systems (the vestibular and tactile systems) implied in the general sensitivity to the movement and the position. These two systems can indeed generate flows independently of the active or passive deformation of the body. Being given this specific property, it is not surprising that Roll [19] suggested the founder role of this system in the emergence of

one experienced body. In addition to the very early functionning of the motor-proprioceptive loop during the foetal life, the question of the basic statute of the proprioception does not seem to be any doubt if one considers work of Held and Hein [8] and especially of Buisseret and collaborators [2]. However, and very basically, it is significant to consider that, on the level more strictly empirical, the description of the constitutive role of the action remains problematic. Indeed, the control condition (absence of action) can be only exceptionally satisfied and with much difficulties what justifies besides the possible recourse to artifacts like robots to validate radically constructivist assumptions. Moreover, it should be considered that this integrating statute of the proprioception is frequently threatened by reafferent co-occurrent flows which can introduce confusions (exteroceptive flow associated the displacement of the environment) or supplant this statute (exteroceptive flow associated to the movement of the body itself which can induce vection illusion). Thus, to pose the funding statute of the proprioception seems admissible only in the terms of a developmental process. We have to note that these important questions are not directly studied in robotic research. Moreover, the movement situates the subject in a temporal unit which resounds on a multitude of natural coupling systems. The unicity of the action is a vector of multimodal integration by way of redundancy, of complementarity. And this point echoes two concepts, suggested by Gibson [7], complementary and extremely relevant to advance in the way of a formalisation of the perceptual learning. These two concepts are those of proprioceptive function and co-perception. Briefly, the first one suggests that the whole of the sensory flows (visual, tactile, vestibular, auditive, etc...) associated to the movement intervene at the same time in the regulation of postural tonicity and the experienced body one. The second one is further posing that, if these flows intervene in the constitution of one oneself (ecological self), they specify simultaneously the world. The proprioceptive function mainly implies low spatial and high temporal resolution (peripheral visual way, tactile spinothalamic way, etc...) sensors of flow. A flow can be defined like the continuous variation of a source of energy on the surface of a sensor. The variation is necessary and it is related to the variation of the sensor position and orientation and/or to the variation of the afferent sensorial flow. The relevant point for the subject is to be able to dissociate these sources of variations; the question of the agentivity defined as the capacity of an agent to perceive that some transformations of the world are directly tied to its proper action is posed at this level. In fact, the organism has signals permanently relating to its positioning in space (deep muscular sensitivity, angular values of the articulations). Exteroceptive flows will be associated, integrated into this major sensitivity. This is the coordination of these two flows which constitutes at the same time the proprioceptive function [3] and co-perception [18]. And the possible detection of temporal coincidences between these two flows constitute the base of the learning of the regularities within the sensory-motor loops (importance of the spatio-temporal redundancies). The formalism that we present at the continuation is inspired very directly by these conceptual evolu-

tions specific to a better understanding of the developmental processes which link motor-sensorial loop, proprioceptive function and perception.

3 Mathematical Definition of Perception

Previous works have focused on mathematical tools to formalize pure behaviorist or reactive systems (see Steels, Smithers, Wiener, Ashby, etc...). The most interesting tools are presented in the classical NN literature and in the dynamical system theory[20]. But these tools are dedicated to specific parts of what we will call a cognitive system (CS)[1]. We summarize here the basis of our mathematical formalism of CS. A CS is supposed to be made of several nodes or boxes associated with input information, intermediate processes and outputs (command of actions). Each element presents a degree of complexity that ranges from a simple scalar product (or distance measure) to a more complex operator such as an "If...then...else..." statement (hard decision making), a recurrent feedback in the case of a dynamical system, a mechanism to control focus of the attention, etc... Whatever the complexity of an element is, we state it as a "neuron". The inputs and outputs of a CS are represented by vectors [2] in the "bracket" notation[3]. An input or output vector noted $|x\rangle$ with $|x\rangle \in R^{+m}$ while its transposed vector is noted $\langle x|$. Hence $\langle x|x\rangle$ represent the square of the $|x\rangle$ norm. We can consider that any element of a CS filters an input vector according to a matrix of weights A and a non linear operator $\mathbf{k}$. This operator represents the way to use the A matrix and the pattern of interactions between the elements of the same boxe. For instance, in the case of a simple WTA (Winner Takes All) boxe, its output $|y\rangle$ is $wta(A|x\rangle)$ with $|y\rangle = (0, ..., y_j, ...0)$ and $j = ArgMax(q_i)$ and $q_i = \langle A_i|x\rangle$. Different kinds of inputs/outputs connections with their weight summarized in the matrix A exist. Basically, we distinguish 2 main kinds of inputs/outputs connections:

- "one to one" connections named I used to transmit information (unconditional stimulus US) from one group to another one, and seen as a reflex pathway which can not be affected by learning.
- "one to all" connections used for transmiting conditional stimuli (CS), having learning capabilities and used for categorization, etc...

Finally in the case of a complex competitive and conditioning structure with 1 unconditional (US) and 2 conditional (CS) inputs, we simply write $|y\rangle = c(A_1|CS_1\rangle, A_2|CS_2\rangle, I|US\rangle)$. This allows to be sure a particular matrix is always associated to the correct input vector but it does not mean the matrix has to be multiplied by the vector (this computation choice is defined by the operator

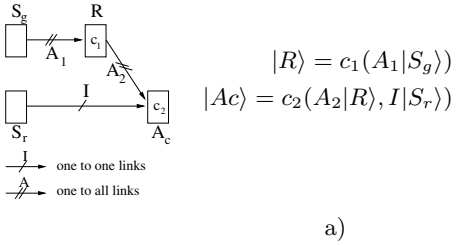
[1] The term cognitive is used in the sense of the study of particular cognitive capabilities (cogitare - to think) and does not induce any a priori cognitivist approach.

[2] We consider the components of the different input/output vectors can only be positive/activated or null/inactivated. Negative activities are banned to avoid positive effects when combined with a negative weight matrix.

[3] The choice of this notation will be explained in the conclusion of the paper.

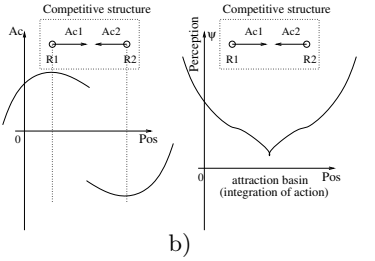
itself). Interestingly this formalism emphasises the fact that an operator is working on 2 different flows of information moving in opposite directions. The first one transforms sensory information in an output code while the second one acts on the group memory in order to maintain an equilibrium defined by the learning rule. Reaching an equilibrium allows the system to have a stable behavior according to its environment but also to adapt itself to environmental variations. These properties echo the psychologic assumptions previously overviewed and show that a percept can only be built if there are interactions between a subject and his environment. Based on these considerations we propose to define the perception as a dynamical attraction basin allowing stable behavior through time. In robotic homing experiments, a similar phenomenon was already observed as learning the construction of a behavioral attraction basin surrounding the goal is enough to allow the robot to return to a place without being able to statically recognise it [6]. Also in visual perception [7], an affordance can be defined as building or accessing to an invariant caracterizing one particuliar sensori-motor behavior. In this case, perception is considered as the result of the learning of sensations/actions associations allowing a globally consistent behavior while facing an object.

For instance, let us consider a sensori-motor system of an agent acting in a given environment (or state space), and having 2 sensation vectors $|S_r\rangle$ and $|S_g\rangle$. First, $|S_r\rangle$ represents the proprioception, a coarse feedback information from the execution of the motor command or the direction of the goal (if the goal is in the immediate neighborhood). It can be considered as a reflex or regulatory pathway linking a proprioceptive sensor to the motor command $|Ac\rangle$. Second, $|S_g\rangle$ represents a more global information about the environment allowing to build a local but robust distance measure (metric). This measure is learned and computed by a competitive recognition group R ($|R\rangle$ representing its output activity). This basic sensori-motor architecture, used as well for a homing task as for a focus on a target, can be described by the diagram fig. 1a and its corresponding equations.



$$|R\rangle = c_1(A_1|S_g\rangle)$$
$$|Ac\rangle = c_2(A_2|R\rangle, I|S_r\rangle)$$

a)



b)

Fig. 1. a) An exemple of a simple sensori-motor system and its corresponding equations. b)*left* Theoretical system actions after learning 2 sensation/action associations and their competition according to the system spatial position. b)*right* Theoretical perception and attraction basin computed by integration of the action shown on left.

The operator c represents a competitive structure (soft-WTA) able to self-organize itself according to one sensory data flow (also in the case of c_2 fig. 1, to condition one input data flow according to an unconditional flow). After the competition, the activity of R reflects the recognition level. The decision is delayed at the motor level and must be understood according to the global temporal dynamic of the system. Let us notice the system behavior does not directly depend on the absolute level of recognition of the learned views or places. Only the rank in the competition process matters which allows some robustness to perturbations until the noise as an effect on the rank in the competition. The output of such a sensori-motor system is an action realized by the agent in its environment. Consequently, the agent modifies its state and then its sensations: the agent and its environment can be viewed as a dynamical system [1]. It is important to notice that in the dynamical systems theory, the action is defined as the derivative of a potential function [10,20]. Considering the psychological background (section 2), we can precise the definition of the perception we gave as an attraction basin: we call this potential function *perception*. Consequently, agent's actions derivate from its state of perception. Formally action $|Ac\rangle$ can be deduced from perception Per with this relation :

$$|Ac\rangle = -m \overrightarrow{grad}\ Per(\boldsymbol{p}) \tag{1}$$

where $\boldsymbol{p}$ is the agent state and m is associated to a "virtual" mass which deals with the agent's embodiment. For instance, m should change while considering two different agents with different morphologies. This mass which allows homogeneity of equation(1) will be considered as a constant in the following. Of course equation (1) can be rewritten: $Per(\boldsymbol{p}) = -\frac{1}{m}\langle Ac|\boldsymbol{p}\rangle = -\frac{1}{m}\int_{\boldsymbol{p}+\delta\boldsymbol{p}} Ac\ d\boldsymbol{q}$. which corresponds to our intuition of the recognition as an attraction basin. An illustration of a perception basin can be viewed on fig. 1b-*right* where only two actions were possible ("go left" and "go right"). The basin results from the numerical integration of the curve proposed fig. 1b-*left* and represents the learned sensori-motor associations and their effects. According to the agent point of view, its perception is built from the integration of its actions relative to its state $\boldsymbol{p}$. In consequence $\boldsymbol{p}$ refers to the agent proprioception and other internal variables which can be implicit in the system. But considering an external point of view, the integration of the agent's actions relative to his spatial position allows to visualize a posteriori a perception which keeps meaning: we obtain a visualization of the dynamical behavioral attraction basin of the agent.

4 Robotic Application

In order to appropriate our definition of perception and understand its full details, we propose a simple robotic application using a Koala robot with a CCD camera. The robot task is to learn how to return to a given object which can be interpreted as the fact the robot "perceives" the object. The robot only learns affordances [7] linked to the target; an explicit recognition of the object is not required. During the experiment, we propose to evaluate the global behavior of

the robot by observing its trajectories. An external observer we will conclude that our robot succeeds in its task if its global behavior is consistent while facing the learned object: a particular affordance is then observed. We also propose to a posteriori compute the robot's perception according to the equation (1). These two observations are in fact complementary: the first one focus on a robot trajectory, which corresponds in a way to descend the perception basin computed by the equation (1).

In order to provide useful and simple sensori-motor associations, the visual features extracted from the visual flow require some robustness as a function of the robot task. As the robot has to move towards an object in unknown environmental conditions, it has to face large non linear transformations of the images (scale, perspectives, etc...). To achieve scale, contrast and luminance invariance, keypoints are extracted on the input images by a multi-scale algorithm inspired by Lowe's work [11]. They fit with the local extrema of the scale-space images filtered by DOGs (Difference Of Gaussians). Finally, at each keypoint, a local feature is coarsly extracted: the first two moments of the orientations of the gradient image in the 4 neighbourhoods of each keypoint are kept. These informations are gathered from all the scales and normalized on neuronal maps which constitute the visual sensory data of the sensori-motor system.
In order to learn object affordances, the robot associates its actions to reach the target object with its sensori data (here visual information). This learning phase is supervised as the direction of the action is provided by an operator via a joystick (fig. 2). This unconditional stimulus corresponds to the "proprioception" pathway reported in section 2. Finally, the association is performed by a conditioning mechanism based on the classical LMS (Least Mean Square algorithm) [23]. The decision about the action is given by a competitive neural group (WTA).

We can notice that the capacities of (linear) separation of such a mechanism are coarse. A simple way to generalize this mechanism in a task involving multiple objects, is to duplicate the group called LMS (for instance one per object selected by the context given by the operator even if it is not a good solution in term of efficiency and biological plausibility). We verify that only a few sensori-motor associations enable the robot to reach the learned object.

As proposed previously, fig. 2 shows one of the robot trajectory recorded after the learning phase was acheived. We can notice that the robot has a direct trajectory when the learned object is in its visual field but zigs zags when the object is not. As the robot doesn't have an explicit object recognition module nor a tracking one, it can't search for the target absent of its visual field. When this happens it just makes its way toward what looks similar to the learned object relative to its perception. Due to the robot movements, the target eventually can be seen again as in fig. 2. Of course the robot visual field is modified by the dynamic of the sensation/action/environment loop.

Finally, we propose to compute the a posteriori perception of the robot. The state of the robot is defined by its spatial location in the environment and by its body and CCD camera orientation relative to the learned object. In order

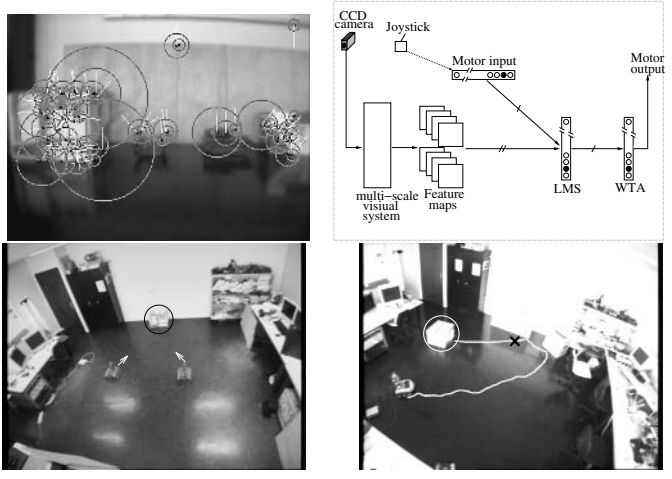


Fig. 2. *Top Left* : Example of an input image and of its keypoints (the circle size indicates the scale where the keypoints were found) *Top Right* : Overview of the architecture with the conditioning mechanism. *Down Left* : Example of a learning phase: the circled object has to be learned. *Down Right* : A robot trajectory (white line). The black cross represents the place where the target (the circled box) enters in the robot's visual field.

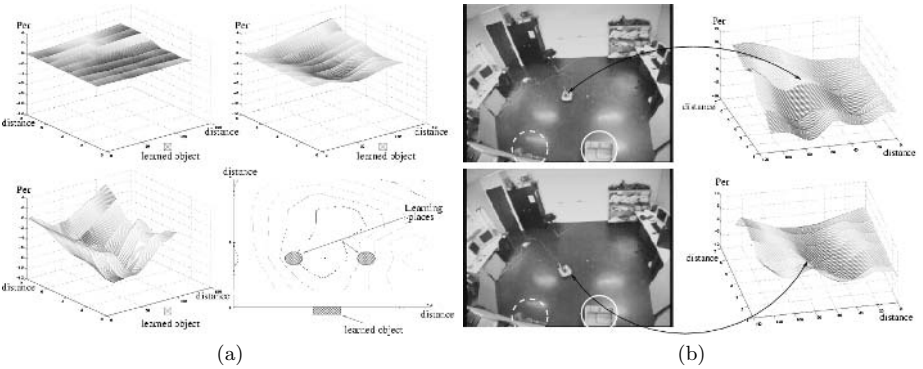


Fig. 3. *(a)* : Visualization of the perception at different learning steps. The different curves of iso-perceptions (right down) associated with the drawing of learning places underline the system capacity to generalize. *(b)* : Visualizations of the perception depending on the orientation of the robot. Top : 2 basins are present even if just the object box (full-line circled) was learned. The object lamp (dot-line circled) creates the second basin. Down: the robot's orientation is modified after a few of its actions. The robot is in a new state of perception without ambiguity

to record the action at each spatial location, we impose the robot's trajectories. Then according to equation (1), perception is given by the integral of the robot action over space. Several measures with a constant orientation of the CCD camera have been successively made during a learning phase. The learning can be considered like the creation of a dynamical attraction basin. The latest is created only after few sensori-motor associations. The shape of the final basin (fig. 3-*a*) fully explains the globally consistent behavior of the robot: the robot "falls" down the perceptive basin and consequently reaches the learning object. These measures corroborate in a quantitative way the results observed on trajectories. Furthermore the good generalization capability of the given architecture can be easily displayed out of the learning places.

The sensory data extracted are coarse and their small number eases the learning of the sensori-motor associations without allowing a good discrimination of objects. While facing the object which was learned previously and an object which was not, neurons activities coding the robot's action are quite similar. Even if the learning allows to dig a deeper basin in the case of the learned object, the difference in depth between the attraction basins we compute is too small to explain the global consistent behavior of the robot: it reaches its learned target independently of its starting spatial location (assuming the learned object is in its visual field). In fact the previous measures do not take into account all the dynamic aspect of the system. In particular, the robot modifies its orientation (and so its sensations) relative to the object according to its state of perception while moving towards the object. These movements allow to disambiguate the visual flow (figure 3-*b*). This dynamic of the sensori-motor loop ensures a consistent behavior which is impossible to obtain in a static way. This experiment clearly shows how essential it is to consider the 3 dimensions of the problem (the spatial location (2 dimensions) and the orientation relative to the object). Only then the notion of a sensori-motor invariant can be grasped but unfortunately the 4D basin cannot be drawn.

5 Conclusion

This paper is an attempt to fill the gap between the psychological concept of perception and the dynamic of a sensori-motor system and the behavior of a robot acting in its environment by proposing a formal definition and an experimental measure of the perception. In this context, learning some particular affordances can be seen as building an attraction basin. A system is in a *stable state of perception* if it is able to maintain itself in the associated attraction basin. Hence recognizing an object (from visual, tactile, auditory... informations) is seen as choosing to stay in a given basin. The choice of the formalism inspired by quantum mechanics (vectors noted with brakets) is linked to the idea that manipulating vectoriel information represents somehow a wave function. An observation of the perception by the agent could be seen as to trigger one particular behavior or in other words to freeze a state of perception (attraction basin). Taking into account this dynamic extends the frame of active vision since the agent becomes an active actor itself. Therefore future works will study how could the agent autonomously construct its own perception of its envronment.

References

1. R.D. Beer. The dynamics of active categorical perception in an evolved model agent. *Adaptive Behavior*, 11(4):209–243, 2003.
2. P. Buisseret, E. Gary-Bobo, and P. Imbert. Ocular motility and recovery of orientational properties of visual cortical neurons in dark-reared kittens. *Nature*, (272):816–817, 1978.
3. A. Bullinger. *Du nouveau né au nourisson*, chapter Vision, posture et mouvement chez le bébé: approche développementale et clinique, pages 48–61. F. Jouen and A. Henocq, Paris, 1991.
4. A.K. Engel and P. König. *Philosophy and the cognitive sciences*, chapter Paradigm shifts in neurobiology: towards a new theory of perception, pages 131–138. Wittgenstein-Gesellschaft, Kirchberg, 1993.
5. P. Gaussier. Toward a cognitive system algebra: A perception/action perspective. In *European Workshop on Learning Robots (EWLR)*, pages 88–100, 2001.
6. P. Gaussier, C. Joulain, J.P. Banquet, S. Leprtre, and A. Revel. The visual homing problem: an example of robotics/biology cross fertilization. *Robotics and Autonomous Systems*, 30:155–180, 2000.
7. J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, 1979.
8. R. Held and A. Hein. Movement produced stimulation in the development of visually guided behavior. *Journal of Comparative and Physiological Psychology*, (56):872–876, 1963.
9. E. Husserl. *Things and space: Lectures of 1907*, volume 7. Springer, 1998. english translation of Dingvorlesung.
10. J.A. S. Kelso. *Dynamic patterns: the self-organization of brain and behavior*. MIT Press, 1995.
11. D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, pages 91–110, 2004.
12. H. Maturana and F.J. Varela. *Autopoiesis and cognition:The realization of the living*. Reidel, 1980.
13. J.K. O'Regan and A. Noe. A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences*, (24):939–1031, 2001.
14. R. Pfeifer and C. Scheier. Sensory-motor coordination: the metaphor and beyond. *Robotics and Autonomous Systems*, 1996.
15. J. Piaget. *La naissance de l'intelligence chez l'enfant*. Delachaux et Niestle Editions, Geneve, 1936.
16. H. Poincaré. *Science and Hypothesis*. Walter Scott, 1905.
17. M. Merleau Ponty. *Phénoménologie de la perception*. Gallimard, 1945.
18. P. Rochat. The self in early infancy. *Theory and research, advances in Psychology*, (112), 1995. Elsevier Science Publishers.
19. J.P. Roll. Physiologie de la kinesthese. la proprioception musculaire: sixieme sens ou sens premier. *Intellectica*, pages 49–66, 2003.
20. G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous System*, 16(2-4):213–245, December 1995.
21. E. Thelen and L.B. Smith. *A dynamic system approach to the development of cognition and action*. MIT Press, 1994.
22. F. Varela, E. Thompson, and E. Rosch. *The Embodied Mind*. MIT press, 1993.
23. B. Widrow and M.E. Hoff. *Adaptive swithing circuits*. IRE WESCON convention records, 1960.

Toward Genuine Continuity of Life and Mind

Liz Stillwaggon

University of South Carolina,
Columbia, SC 29208

Abstract. The *strong continuity thesis* was introduced into the artificial life literature in 1994, [5], but since then has not received the attention and further development it merits. In this paper, I explain why if we are to identify genuine continuity between life and mind, a shift in perspective is needed from thinking about living and minded things and processes, to thinking about Life itself and Mind itself. I describe both life and mind as self-preserving processes and argue that this notion accounts for their purported continuity, drawing on research in embedded and embodied cognition to make my case. I then respond to Peter Godfrey-Smith's observation that any view on which thought requires language is inconsistent with the strong continuity thesis by arguing that although such a view of thought might be rendered consistent with the thesis, a dynamic systems approach to cognition, i.e., one wherein thought is language-*in*dependent, is much more conducive to identifying genuine life-mind continuity.

1 Introduction

Just over ten years ago the thesis of *strong continuity* between life and mind was introduced into the growing literature on artificial life [5]. This continuity thesis was not developed in any depth then but was simply explained as follows: life and mind differ in degree and not kind, thus the functional properties of mind are an enriched version of the functional properties of life. Furthermore, since life and mind have in common an abstract pattern or set of organizational properties, "mind is literally life-*like"* [5]. These basic theoretical tenets suggest a fundamental principle that has both ontological and methodological significance, namely that since mind is fundamentally like life, our efforts to understand the mind are continuous with our efforts to understand life in general. This paper explores what evidence there is for accepting such a principle.

The first task I take on in this paper is to identify what it is about life and mind that could account for their purported continuity. Andy Clark has noted that perhaps the most difficult task in naturalizing the mind in the way prescribed by the strong continuity thesis is in finding a balance between the explanatory needs of seeing continuity in nature while somehow still recognizing that the mind is special [3]. Evolutionary theory tells us that consciousness evolved much later than did life itself, which suggests at least developmental and historical continuity between living things and 'minded' things, but I believe the significance of the thesis in question lies in its trying to identify continuity at a much more fundamental level. The appropriate level of analysis is not living things and minded things, but rather Life itself and Mind

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 47 – 56, 2005.

itself; our concern is with the two like phenomena rather than with the entities that instantiate such phenomena. Research in embedded and embodied cognition is most amenable to identifying continuity between life and mind, so work in this area is drawn on to help elucidate the contours of the strong continuity thesis.

The second task in the paper is to revisit the original (brief) formulation of the strong continuity thesis and respond to a curious assumption made about it by the author. Peter Godfrey-Smith takes all views on which thought requires language to be views that are necessarily inconsistent with the strong continuity thesis. His intuition here apparently is that human language is too different from the principles of life for the two phenomena to be rendered conceptually consistent. But it seems that in order for the continuity thesis to be fully comprehensive, it must either provide an account of language-dependent thought, or show why such a view of thought is mistaken. I argue that although either model of thought might conceivably be rendered consistent with the strong continuity thesis, the latter option, i.e., that thought does not require language, is much more conducive to genuine life-mind continuity. This claim is supported with insights from dynamic systems theory.

2 (Not Much) Background on the Strong Continuity Thesis

Since publication of the article in which appears the strong continuity thesis, only passing references have been made to it in the literature. I do not believe this lack of discussion results from a corresponding lack of interest in the subject, but rather is due to the fact that the identity the thesis postulates between the composition of mind and the composition of life is a notion implicitly accepted by the artificial life community, perhaps most clearly expressed in its attempts to distinguish itself from the older (and some believe, failed) AI program. Christopher Langton expresses just this point when he says, "From the very beginning artificial intelligence embraced an underlying methodology for the generation of intelligent behavior that bore no demonstrable relationship to the method by which intelligence is generated in natural systems ([6], p.41)." Since AL *is* concerned with simulating natural systems, its programs by contrast are designed to generate *lifelike* behavior which may or may not produce behavior we would deem 'intelligent'. Strongly influenced by ideas from cognitive science and dynamic systems theory, A-Lifers recognize that the organism-plus-environment creates a unified, living system, and that to abstract intelligence away from this dynamic is to completely miss the big picture.

It is a theme of cognitive science, and to some extent of AL, that intelligent behavior emerges given a certain level of complexity within systems, whether natural or artificial. Rodney Brooks draws on evolution to make the point that while it took forever for *living* systems to develop, it took comparatively far less time for *thinking* systems to do so; "Problem-solving behavior, language, expert knowledge and application, and reason are all pretty simple once the essence of acting and reacting [is] available ([2], p.396)." The significant insight expressed here is that life preceded mind, and although one might postulate that their order is contingent, in other words that mind *could have* preceded life, the fact of the matter is that life is necessary for mind, that mind could not exist without life. Although this notion amounts to a direct and necessary kind of continuity between life and mind, it is nevertheless a weaker

notion than what is called for by the strong continuity thesis, and is in fact what Godfrey-Smith calls the *weak thesis* [5], namely that anything that has a mind is alive (although the converse, of course, does not hold). From these cases, I believe it is evident that the sentiment concerning life-mind continuity is already in the literature, thus what is needed is a more explicit account of it.

3 The First Step: Defining Life

The life-mind continuity sought is, I take it, indicative of the larger project to naturalize the mind, that is, to explain the 'mystery' of consciousness by appealing to nothing but the familiar principles of life of which we have a pretty good understanding. As Clark puts it, "the thesis of strong continuity would be true if, for example, the basic concepts needed to understand the organization of life turned out to be…*those very same concepts and constructs*…central to a proper scientific understanding of mind ([3], p.118)." Of course what is wanting is an identification of those particular concepts or constructs that account for the purported life-mind continuity and, to my knowledge, no arguments of this kind have yet been put forth. We are not completely in the dark however; the extensive literature on the related question of how to define and understand life itself should provide helpful insight into finding the most promising candidates to account for the continuity between life and mind, since after all the claim is that the very same principles will be central to an understanding of both.

Claus Emmeche, Mark Bedau, and others have noted the preponderance of definitions of life including biochemical, thermodynamic, physiological, metabolic, and genetic definitions, all of which identify what seems to be a necessary characteristic of life, yet none of which identifies a sufficient one [4]. For example, Emmeche explains how the metabolic definition fails to define life uniquely because it states that, "a living system is one that is distinct from its external environment and that exchanges material with its surroundings…without changing its general properties," which, he points out, also accurately describes a whirlpool in a river ([4], p.34)." The other definitions suffer the same fate of applying not only to living systems like organisms but also to complex systems that are not living. It is reasonable, based on these musings, to propose (and some have) that life may be a cluster concept, best understood as a unique *state* or *process* emerging from a special combination of requisite characteristics. And based on the particular continuity sought by our thesis, mind too then would have such a multifarious nature.

A relatively early and very influential account of life comes from Maturana and Varela and their notion of 'autopoiesis' [7]. I understand this term to mean roughly 'self-reproduction', so that a system of this kind is continuously making and remaking itself by maintaining its own boundaries and carrying out complex interactions within to produce essential materials. Bedau's notion of *supple adaptation* is closely related: "Individual living entities (organisms) maintain their self-identity and their self-organization while continually exchanging materials, energy, and information with their local environment ([1], p.332-333)." Supple adaptation is to be distinguished from simple interaction between an entity and its environment, as in the case of a thermostat's calibration with a room's temperature, or the constancy of a whirlpool in

a river, in that "supple adaptation involves responding appropriately in an indefinite variety of ways to an unpredictable variety of contingencies ([1], p.338)." This notion of continual interaction with and adjustment to the environment serves to uniquely mark out living from nonliving systems.

4 Toward Genuine Continuity

I believe that trying to define life in terms of its multifarious characteristics is an inherently flawed project because of what it implies about the nature of life. All of the above definitions of life (metabolic, genetic, etc.) identify not what life itself *is*, but rather what living entities *do.* Organisms metabolize nutrients, reproduce, etc., and if what one seeks is a better grasp on the determining characteristics of being alive, then such definitions are useful. However, such an approach does not get us any closer to identifying continuity between life and mind because even if we were able to define life once and for all in terms of its necessary characteristics, this would only then leave us trying to identify those same characteristics in the mind, and such efforts to bridge the gap, so to speak, between life and mind could only ever succeed in identifying likenesses or similarities, and not genuine continuity. This is because, for example, my car's being blue and the ocean's being blue does not constitute some sort of car-ocean continuity; instead what is established is merely their having one feature in common. But what if life were a cluster concept, realized only when the requisite dozen characteristics were present? Still, the objection holds. There are many features common to all of the major cities in the United States, for instance, but this of course does not establish genuine *continuity* among them. Something more than mere feature sharing will account for life-mind continuity if anything will. If what is sought is an understanding of the phenomenon of life itself, with the intention of at the same time coming to a better understanding of mind itself, then it seems that a change in perspective is called for.

a. Life as Process

It is a fundamental intuition of AL that there may be more to life than our earthbound biology has been able to teach us. It has been suggested that life is a process potentially instantiated by a variety of complex, self-organizing systems in different media. A-Lifers seems to think of life in more fluid terms than do biologists, emphasizing that life is an attribute of the *organization* of particulates and not of the particulates themselves. There are reverberations of this notion of life-as-process even in the traditional sub-disciplines of biology. Talk can be heard in genetics, for example, of patterns of genes, rather than individual genes, effecting phenotype; the focus has shifted from trying to identify a single gene for eye color to instead recognizing that blue eyes may result from a set of complex interactions among several genes. Following this line, phenotypic traits are better conceived as familiar patterns in the overall process of life. I mention all of this only to suggest that conceptualizing life as the sum total of various processes is not radically new; what *is* radical, however, is the notion of life in general being a process, which breaks with the tradition of thinking about life in terms of living *things* and forces us to think of the phenomenon of life itself.

What bearing would such a conceptualization of life have on the notion of continuity between life and mind? Clearly, it would imply that the two phenomena were themselves processes; the focus would shift from *life processes* like metabolism and reproduction, and *mental processes* like imagining and remembering, to conceptualizing life and mind as processes in and of themselves. Put another way, while the current definitions of life and mind presuppose the two phenomena to be things with attributes, the shift in perspective I am proposing would instead have us think about life and mind as processes with characteristic natures.

But, one should be thinking at this point, *lots* of things count as processes—making coffee and doing one's taxes for example—yet no one would argue that, as processes, these activities therefore have something fundamental in common with life and mind. Fair enough. But instead of abandoning the notion of conceptualizing these phenomena as processes, I propose that there is something that distinguishes them from all other processes, and identify the fact that they are self-preserving processes as this crucial something. What does it mean to be 'self-preserving'? The notion, I would concede, is very similar to that of autopoiesis or supple adaptation, introduced above, wherein the organism preserves itself as distinct from its environment through mechanisms characteristic to it. What *is* different, however, is the context; I want to apply the notion of a self-preserving process, intrinsic to many definitions of life, to mind, and thus identify their purported continuity.

Many things are non-processes; a book keeps on being a book in virtue of nothing it does but rather in virtue of the laws of physics that 'allow' it to maintain its given form. And likewise many things are processes yet not self-preserving ones; the process of making coffee is *not* self-preserving because it does not occur independently of an agent (excepting of course the timed, automatic coffee makers that some of us are lucky enough to own!). But life (and below I will argue, mind) are not so; they preserve themselves in virtue of various necessary sub-processes that keep working for the duration.

I think it is straightforward enough to conceptualize life as a self-preserving system; multiple examples can be drawn from the cellular, organismic, and (possibly even) special level. Living biological cells spontaneously participate in a whole host of processes that keep them alive, likewise with organisms (i.e., all those already discussed, e.g., metabolism, etc.), and species, that engage in reproduction and other less obvious processes that keep their genetic lineages alive. I like the idea, motivated by AL insights, that Life may be an abstract process, potentially instantiated by the multifarious life forms we know from biology, the others we create in silico, and still others we invent in thought experiments and science fiction stories. Perhaps shifting our focus from one of trying to learn about the features of living things, to the nature of the abstract process of which they are all instantiations, will allow us to arrive at a better, more fundamental, understanding of life.

b. Mind as Process

But even if we can conceive of life as a self-preserving process, do we have any reason to believe that the mind may be of the same nature? Yes, I believe we do. Cognitive features like intelligence, learning, and memory can easily be conceived as ones that facilitate survival, by helping one navigate one's environment successfully for example, an ability dualistically measured as intelligence in humans and

functionality in robots. But, one might say, don't abilities like these simply amount to the preservation of one's body, and not one's mind? Yet even to acknowledge this objection as a problem for our view is to abandon it in spirit; since we are trying to establish life and mind as being fundamentally continuous phenomena, it does little good to pose questions that presuppose dualism. The mind needs the body to survive; put metaphorically, the body is a vessel that keeps the mind afloat. While we do have empirical evidence of mindless bodies (think either of humans in a permanent vegetative state or any non-human animal that is not purported to have a mind), we have none of the reverse. True, bodiless minds abound in talk of gods, spirits, and ghosts, but herein I am concerned only with identifying continuity between the kind of mind with which we are familiar, namely our own, and the kind of life (whether natural or artificial) with which we are also familiar.

What empirical evidence do we have to support the claim that mind is a self-preserving process? A distinction was emphasized above between the older AI paradigm of simulating abstract, intelligent behavior, and the newer paradigm in AL of generating lifelike systems that may or may not behave 'intelligently'. The key claim resulting from embedded and embodied cognition studies is that intelligence in humans (and functionality in robots) emerges dynamically as a function of the degree of interactiveness between creature and environment; in both cases success is measured by how well the subject navigates and/or manipulates its environment. A good example of such work comes from Brooks and his research team at MIT who created the robots Allen and Herbert. These robots are antithetical to former AI projects in several ways: they reside in the real world rather than inside a computer, they are controlled by simple rules and virtually no memory so that reliance on an internal cognitive map is out of the question, and they successfully 'rise to the occasion' when faced with unexpected challenges. In sum, they are the embodied examples of 'intelligent' systems whose "*intelligence is determined by the dynamics of interaction with the world* ([2], p.418)." The world (albeit mindlessly) supplies the robot with information, for example where obstacles are, and the robot continually reacts to that information and adjusts its direction accordingly.

An insight we can draw from such studies is that the mind preserves itself, indirectly as it were, by preserving the body that hosts it. Of course explaining it in this way inadvertently implies a dichotomy between the living organism and the mind to which it is host but this need not trouble us—the strong continuity thesis allows that the mind is an enriched version of life and that the two differ in degree; the claim is not that life and mind are numerically or otherwise identical but rather that their natures substantiate a genuine continuity between them.

So far we have looked at the notion of mind's preserving itself via its direction of the body. But if we want to postulate mind as a self-preserving process, need we identify reasons why the mind should want to preserve itself for itself? I do not think so. While the survival instinct is often cited as an explanation for self-preservation behaviors in animals, our analysis focuses on Life itself, and no simple explanation can be given for why Life in general sustains itself, it just does. Rather than an entity driven to self-preservation in and of itself, mind, being an enriched version of life, is better conceived as a sophisticated facet of life's preserving itself. Human intelligence is a measure of how well we function in our environment; while an earthworm has only to dig, eat, and excrete, the multifarious environment that is home to humans is

demanding in much more complex and complicated ways that necessitate a flexible and adaptive mind.

c. A Note on AL and AI.

It has already been mentioned that it is a notion familiar to the AL community that life may be an abstract process instantiated by either natural or artificial systems, so long as the appropriate complexity (and other conditions) are met. It should also be noted that the notion of mind as process is familiar to some: Clark borrows Marvin Minsky's unpleasant phrase for the brain, i.e., 'meat machines', to entitle the first chapter of his book *Mindware* [3]. Clark explains the basic idea: "Mindware, it is claimed, is found "in" the brain in just the way that software is found "in" the computing system that is running it ([3], p.8)" and goes on to say, "the brain may be the standard…implementation—but cognition is a program-level thing ([3], p.13)." The relevant comparison herein is not between brain and computer, but cognition and program—the focus is on the behavior of the system rather than on the system itself, and both computer programs and cognition are processes rather than things.

Also, a case could be made for Alan Turing's research being amenable to the idea of mind as process. He was interested in the abstract concept of 'machineness' (my term), i.e., the idea that the function performed by the machine mattered far more than its material instantiation, captured well by his thought experimental Turing Machine [10] (as it came to be called). He was concerned with the question of whether machines could think in a way like humans (indeed, enough like us to fool us, i.e., the famous Turing Test). Although Turing's language did not include references to mind's being a process, it is a notion that does not seem inconsistent with his research into the similarities between machine functionality and human thought.

5 Avoiding Discontinuity

I turn to the second task of the paper. In his 1994 paper, Godfrey-Smith mentions that he takes all views on which thought requires language to be views that are necessarily inconsistent with the strong continuity thesis. I believe Godfrey-Smith is appealing to intuition (his own and the reader's) to make the point that although thought *may* require language, it is too foreign a notion to suggest that life too could require language, and furthermore that if thought does *in fact* require language, this leaves no room for life-mind continuity. Although I am sympathetic to the intuition here, it nevertheless is one that begs for further explanation; if thought *were* language-dependent, could it still in some way be made consistent with the strong continuity thesis? Alternately, is a different model of thought more conducive to genuine life-mind continuity?

If the notion of thought's being language-dependent can be justifiably reduced at least for the sake of analysis to its being symbol-dependent, the result might look something like Newell and Simon's *physical symbol system hypothesis* [9]. This hypothesis, quite powerful in its time, was criticized by the later wave of embodied cognition, which emphasized the importance of the unique dynamic created by creature and environment to generating intelligent behavior. The following quotation from Newell and Simon (1976) is telling: "Thought was still wholly intangible and

ineffable until modern formal logic interpreted it as the manipulation of formal tokens ([9], p.108).” Thought is symbol manipulation, pure and simple. Although I find this explanation of thought problematic and limited, I will grant it for the sake of argument because the question at hand is whether continuity between life and mind *could be* identified given the structure of thought entailed by the PSSH. To answer this question, we need to take a look at the other side of the equation: life.

Can life be conceived as a form of symbol manipulation? It seems the most promising route to developing such a thesis would come from genetics; DNA and RNA are strings of symbols responsible for exchanging essential information and directing the development of the organism. So perhaps then thought and life are both instances of symbol manipulation and information exchange? There are two main reasons why such a thesis strikes me as potentially problematic: 1) the notions of symbol manipulation and information exchange are too broad—too many nonliving and non-minded entities fall into these categories; and 2) such an approach is an instance of trying to identify shared features, objected to earlier—symbol manipulation may describe certain types of thought, but not mind itself, and likewise certain life processes, but not life itself.

There are two notions from research in dynamic systems theory I would like to draw on to argue why a non-computational model of thought, i.e., one wherein thought does *not* require language, provides a much more promising route to the requisite, genuine continuity between life and mind. The first I will call ‘transparency’, as distinct from representation. The central idea has already been referred to throughout the paper and is nicely summed up in the title of Brooks’ 1991 article, “Intelligence Without Representation” [2]. The basic idea is that intelligent behavior need not result from abstract symbol manipulation, but rather can emerge from basic interactiveness with one’s environment. Brooks tested this model of cognition empirically with his robots Herbert and Allen and claims that the key idea is “to use *the world itself as its own best mode,* ([2], p.405).” These robots were CPU-less so they had no central memory store by which to encode a cognitive map of their surroundings—their success was due entirely to their real-time interaction with their environment. Another well-known example of transparency comes from van Gelder’s argument that the same function can be performed by a standard Watt governor as a computational one [11]. What follows by implication is that although a PSSH may seem to capture certain features of thought, or some might say, thought itself, it is most likely another example of mistaking flight for mere wing-flapping [2]. Conceptualizing thought as symbol manipulation does seem to capture certain cognitive functions, like doing logic problems and writing sheet music for example. But it is not clear how such a model of thought could ever adequately describe other abstract, presumably non-computational, cognitive functions like humming a tune or thinking about a friend, much less how it could describe basic cognitive functions like maintaining balance, clapping hands, or smiling.

The second notion from dynamic systems theory relevant to developing a non-computational, that is, language-independent, model of thought, is what Clark calls “cognitive incrementalism” ([3], p.135), and Brooks calls “incremental intelligence” ([2], p.401). Although the authors might want to distinguish the two notions (e.g., Clark’s notion is more theoretical and Brooks’ is empirical), for our purposes it suffices to understand these related notions as the idea that more involved and

difficult cognitive abilities build on more basic ones, so the type of mental processes that account for simple tasks like breathing and walking are continuous with those that account for abstract thinking and planning. Such a model of thought presupposes the importance of the subject's being embodied and embedded in its surroundings; bottom-up approaches that generate behavior we recognize as 'intelligent' are favored for being more realistic than top-down approaches that simulate bodiless minds excelling at chess or other abstract tasks.

The concept of cognitive incrementalism more or less shadows the concept of strong continuity between life and mind. It was stated in the beginning of the paper that the few theoretical tenets of the strong continuity thesis, e.g., that the functional properties of mind are an enriched version of the functional properties of life, and that therefore mind is life-like, suggest a methodological principle, namely that if the thesis is right, our sciences of the mind should be continuous with our sciences of life in general. Empirical work in the field of embedded and embodied cognition has produced the most promising evidence for the validity of cognitive incrementalism and thus for the validity of the strong continuity thesis. Such studies show rather compellingly that intelligence may be much more transparent a concept than we once believed, mistakenly conceived at first as abstract and immaterial until securely grounded in reality by the realization of how natural systems *actually* exhibit intelligence. Conceptualizing mind *not* as an abstract symbol system but rather as life preserving itself through the most effective means possible is conducive to recognizing genuine continuity between life and mind.

6 Conclusion

The field of AL has made many contributions to several disciplines but its most important contribution to philosophy has been its insistence, from the outset, on challenging our familiar conceptualizations about life and mind. AL forces us to question long held definitions and distinctions often founded in deep tradition, and is for this reason, I believe, sometimes met with resistance from philosophy. However, it is in the true spirit of philosophy to allow one's picture of the world to come apart at the seams and then still try and make sense of it all. Insights from AL and AI provide some promising avenues to reconceptualizing the phenomena of life and mind in ways conducive to identifying their continuity and herein I have given one such account.

A worthwhile project would be to explain how the fields of AL and AI are themselves conceptually continuous and thus flesh out more interesting ways in which our sciences of life and our sciences of the mind are methodologically continuous. One philosophical question that might be enlightened by such a project is whether personhood or consciousness is abstractable from the body in the same way that a program is abstractable from the machine. Turing's conceptual 'universal machine', and all contemporary computer programs, provide good evidence for the claim that 'machineness', i.e., the essence of the machine, is indeed an abstractable quality. Yet when we put the question to ourselves, i.e., whether we are dependent on, or identical to, our bodies, or whether our being is abstract, like a soul or consciousness that just happens to be instantiated by our biological bodies, the answer is far from clear.

Probably AI's most important contribution to philosophy is that it enables us to objectify what is so elusive and opaque when we try to understand it subjectively, namely the human mind. The metaphor I like to use for AI is that of erecting a mirror that reflects back to us what was difficult, if not impossible, to see without it. Undoubtedly AI will continue to provide insights into the human mind, and if these insights are appreciated as continuous with those from AL, it may provide us with a deeper understanding of the continuity between life and mind that would have been impossible to achieve otherwise.

References

1. Bedau, Mark (1996). "The Nature of Life," in Margaret A. Boden (ed.), *The Philosophy of Artificial Life* Oxford: Oxford University Press, 332-357.
2. Brooks, Rodney (1991). "Intelligence Without Representation," in John Haugeland (ed.), *Mind Design II,* Cambridge, MA: MIT Press, 395-420.
3. Clark, Andy (2001). *Mindware.* New York: Oxford University Press.
4. Emmeche, Claus (1994). *Garden in the Machine.* Princeton: Princeton University Press.
5. Godfrey-Smith, Peter (1994). "Spencer and Dewey on Life and Mind," in Margaret A. Boden (ed.), *The Philosophy of Artificial Life* Oxford: Oxford University Press, 314-331.
6. Langton, Christopher (1996). "Artificial Life," in Margaret A. Boden (ed.), *The Philosophy of Artificial Life,* Oxford: Oxford University Press, 39-94.
7. Maturana, Humberto, and Varela, Francisco (1980). *Autopoiesis and Cognition.* Dordrecht, The Netherlands: Reidel Press.
8. Minsky, Marvin (1994). "Society of Mind: A Response to Four Reviews," in W. Clancey, S. Smoliar, and M. Stefik (eds.), *Contemplating Minds.* Cambridge, MA: MIT Press, 308-334.
9. Newell, Allen, and Simon, Herbert (1976). "Computer Science as Empirical Inquiry: Symbols and Search," in John Haugeland (ed.), *Mind Design II.* Cambridge, MA: MIT Press, 81-110.
10. Turing, Alan M. (1950). "Computing Machinery and Intelligence," in B. Jack Copeland (ed.), *The Essential Turing.* Oxford: Clarendon Press, 433-464.
11. Van Gelder, Timothy (1996). "Dynamics and Cognition," in John Haugeland (ed.), *Mind Design II,* Cambridge, MA: MIT Press, 421-450.

Biological Development of Cell Patterns: Characterizing the Space of Cell Chemistry Genetic Regulatory Networks

Nicholas Flann[1], Jing Hu[1], Mayank Bansal[1],
Vinay Patel[1], and Greg Podgorski[2]

[1] Computer Science Department, Utah State University, Logan UT, USA
[2] Biology Department, Utah State University, Logan UT, USA
nick.flann@usu.edu
http://www.cs.usu.edu/~flann/index.html

Abstract. Genetic regulatory networks (GRNs) control gene expression and are responsible for establishing the regular cellular patterns that constitute an organism. This paper introduces a model of biological development that generates cellular patterns via chemical interactions. GRNs for protein expression are generated and evaluated for their effectiveness in constructing 2D patterns of cells such as borders, patches, and mosaics. Three types of searches were performed: (a) a Monte Carlo search of the GRN space using a utility function based on spatial interestingness; (b) a hill climbing search to identify GRNs that solve specific pattern problems; (c) a search for combinatorial codes that solve difficult target patterns by running multiple disjoint GRNs in parallel. We show that simple biologically realistic GRNs can construct many complex cellular patterns. Our model provides an avenue to explore the evolution of complex GRNs that drive development.

Keywords: GRN, cascading GRNs, recurrent GRNs, artificial embryology, development, developmental programs, cellular differentiation, differential equations, hill-climbing.

1 Introduction

Development from a single cell, the zygote, into the adult organism is a remarkably complex and poorly understood process. One common way to create patterns in the early embryo is through the use of diffusible morphogens that form gradients to provide positional information to embryonic cells. Cells acquire different identities within a developing field according the levels of morphogen they detect. Broad divisions within the embryo can be established this way. More elaborate and finely resolved patterns are often created later when cells that have acquired coarse positional identities interact across membranes. These interactions generate different cell types in arrangements that include borders, patches and mosaics. Development in *Drosophila* follows this pattern, beginning with morphogens that specify broad divisions along the anterior-posterior and

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 57–66, 2005.

dorsal-ventral axes, followed by cellular interactions that create sharply bounded divisions [18].

Many investigators have developed computational models that reproduce elements of development[20][19][13][15][5]. This is a relatively young field with great potential to provide insights that complement and extend those obtained by classical biological investigations. An important approach to modeling development is Artificial Embryogeny (AE). One path in AE follows a grammatical approach where sets of rules in the form of grammatical rewrite systems are evolved [1][2]. The other path of AE research is a cell chemistry-based approach that simulates the way structures emerge in biology e.g. [1][8][9][10][6].

Inspired by nature's elegance and precision in solving the problem of embryogenesis, we have developed a cell chemistry-based AE system to search for biologically realistic GRNs capable of generating patterns found in embryos. We study the effectiveness of GRNs in solving target patterns of cell types such as borders, patches, and mosaics that are observed in biological development. The power of this GRN search is that it allows unrestricted exploration of ways to solve embryological patterning problems. This stands in contrast to the actual embryo which, while solving complex problems of patterning, is a prisoner of its evolutionary history.

2 Approach

2.1 Cell Pattern Problems

Our focus is on how GRNs can build 2D spatial patterns similar or identical to those generated during biological development. A sample of the 43 patterns used in this study is shown in Fig.1. The individual patterns are grouped into three broad types: mosaic, border and patch. A mosaic pattern is defined as a periodic pattern in one or two dimensions repeated across the sheet of cells. A border pattern identifies single cells or single lines of cells that divides one group of cells

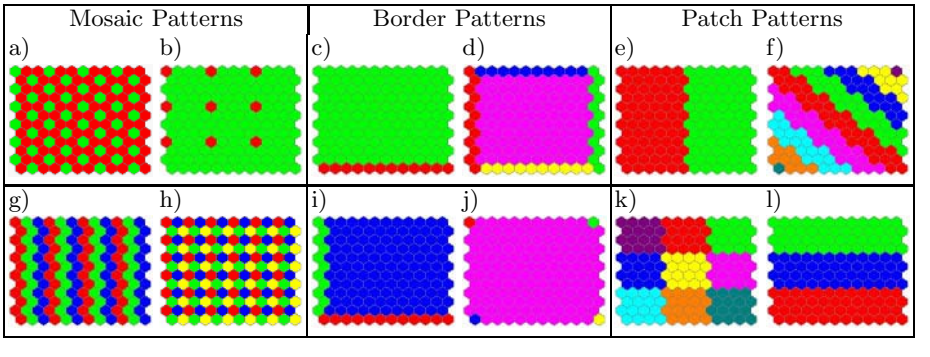


Fig. 1. A sample of the 43 target patterns used in this computational study. Each problem is represented as a 2D sheet of hexagonal cells (12 by 12, or 24 by 24). The complete set of target patterns is given in supplementary material [27].

from another or from the medium. A patch pattern is an aperiodic partitioning of the sheet of cells into contiguous groups of distinct cells types. Segments are a biologically significant subset of the patch pattern. Each of the pattern types is common in natural development. For example, regular mosaics occur in *Drosophila* epidermal neurogenic clusters where a single neuroblast becomes surrounded by non-neuronal supporting cells [18] and in the vertebrate retina where ganglion cells are encircled by cells of other types [24]. Patches in the form of segmental repeats are seen in the obvious segmental divisions along the insect anterior-posterior axis [18] and in the segmental array of vertebrate somites [26]. Borders are seen frequently, such as in the strip of cells that becomes a signaling center at the boundary between the anterior and posterior compartments of the *Drosophila* wing imaginal disc [18] and in the apical ectodermal ridge at the division between the dorsal and ventral ectoderm of the vertebrate limb bud [25].

2.2 Modeling Biological GRNs

Space of GRNs. Biological models of GRNs can be described as a graph, where each node represents a distinct protein's expression level and each edge represents influences among proteins. A protein is influenced when its production or inhibition is controlled as a function of other protein expression levels. Since production and inhibition are defined as rates of change, the GRN is naturally modeled as a set of coupled differential equations. Fig.2 shows an example of a 3 protein, 5 edge GRN and its input and output protein expression pattern.

Table 1. The edges composed to form a GRN. The rate of change of protein P_0 for cell σ is defined in terms of the weighted *expression* of protein P_i in σ and neighboring cells, where ω_j is the strength of edge j $(0.0 \leq \omega_j \leq 1.0)$, $f(x) = \frac{x^2}{(1+x^2)}$; $g(x) = 1 - f(x)$; $h(x) = \frac{2}{(1+e^{-x})} - 1$; $n(\sigma)$ returns the set of directly neighboring cells (i.e., that share a common membrane); n_S, n_W, n_N, n_E return the cell directly neighboring to the South, West, North and East respectively.

Label	Description	Name	Definition
A	P_0 Diffusion with zero boundary conditions	***Diff0***	$\frac{dP_0(\sigma)}{dt} = \omega_j \frac{\partial^2 P_1(\sigma)}{\partial x^2} = \quad P_1(\sigma)$
B	P_0 Diffusion with fixed boundary conditions	***DiffX***	$\frac{dP_0(\sigma)}{dt} = \omega_j \frac{\partial^2 P_1(\sigma)}{\partial x^2} = \quad P_1(\sigma)$
C	P_0 direct expression by P_1	***ExprDirect***	$\frac{dP_0(\sigma)}{dt} = \omega_j f(P_1(\sigma))$
D	P_0 direct inhibition by P_1	***InhDirect***	$\frac{dP_0(\sigma)}{dt} = \quad \omega_j f(P_1(\sigma))$
E	P_0 driven to same as P_1	***Same***	$\frac{dP_0(\sigma)}{dt} = \omega_j (f(P_1(\sigma)) \quad P_0(\sigma))$
F	P_0 driven to opposite of P_1	***Oppos***	$\frac{dP_0(\sigma)}{dt} = \omega_j (g(P_1(\sigma)) \quad P_0(\sigma))$
G	P_0 driven to difference in values between P_1 and P_2	***Error***	$\frac{dP_0(\sigma)}{dt} = \omega_j (h(P_1(\sigma) \quad P_2(\sigma)) \quad P_0(\sigma))$
H	P_0 autocatalysis and reciprocal control by P_1	***ExprQuad***	$\frac{dP_0(\sigma)}{dt} = \omega_j (f(\frac{P_0(\sigma)^2}{P_1(\sigma)} + \psi_j))$
I	P_0 quadratic inhibition by P_1	***InhQuad***	$\frac{dP_0(\sigma)}{dt} = \omega_j (g(P_1(\sigma)^2) \quad \psi_j)$
J	P_0 driven to same as cell neighbors values of P_1	***SameNeig***	$\frac{dP_0(\sigma)}{dt} = \omega_j (f(\sum^{\rho \quad n(\sigma)} \frac{P_1(\rho)}{6}) \quad P_0(\sigma))$
K	P_0 driven to opposite of cell neighbors values of P_1	***OpposNeig***	$\frac{dP_0(\sigma)}{dt} = \omega_j (g(\sum^{\rho \quad n(\sigma)} \frac{P_1(\rho)}{6}) \quad P_0(\sigma))$
L	P_0 driven to difference in opposing cell neighbor values of P_1	***ErrorNeig***	$\frac{dP_0(\sigma)}{dt} = \omega_j (f(\sum^{\rho \quad n(\sigma)} \frac{P_1(\rho) \quad P_1(op(\sigma,\rho))}{6}) \quad P_0(\sigma))$
M:P	P_0 driven to same as geometric neighbor value of P_1; with i N, W, S, E	***SameNeig***$_i$	$\frac{dP_0(\sigma)}{dt} = \omega_j (f(P_1(n_i(\sigma))) \quad P_0(\sigma))$

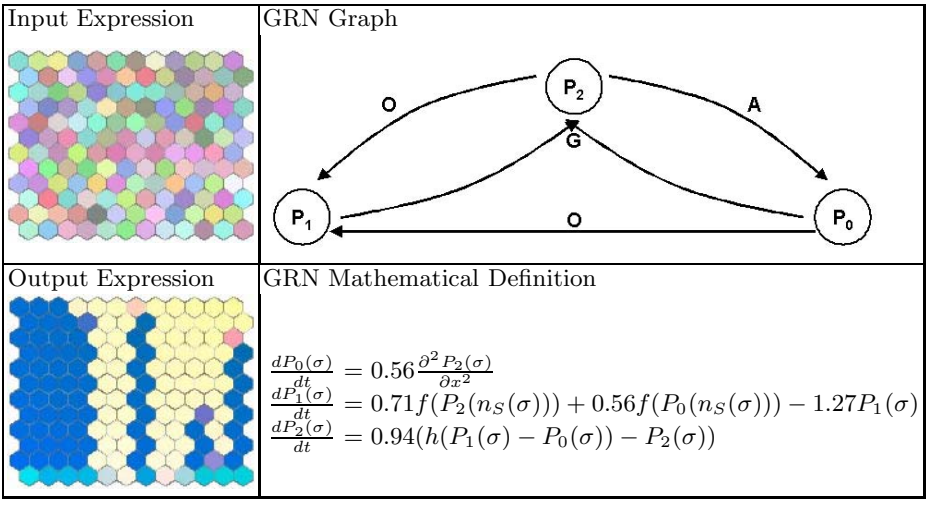


Fig. 2. Example of a GRN. The left column shows the input and output protein expression patterns, where the color of each cell is computed by mapping protein expression directly onto the *RGB* values. The right column shows the GRN as a graph and the associated set of coupled differential equations. The letters indicate edge types shown in Table 1.

Table 1 illustrates the protein influences considered in this study. Within an individual cell, protein expression can be controlled by a single protein (the direct control edges *C*-*F*, *I*) or some function of multiple proteins (the combinatorial control edges *G*, *H*). Over the sheet of biological cells, proteins influence each other through both long-range and short-range signaling. Edges *A* and *B* implement long-range signaling through diffusion under different boundary conditions. Edges *J*-*P* implement short-range signaling, where a cell can sense protein expression levels in directly neighboring cells across contacting membranes [20]. Notice edges *M*-*P* enable a cell to signal to a specific geometric neighbor cell. Such capabilities are known to be utilized to build internal segment borders [13] and rely on morphogenic gradients constructed earlier that establish anterior-posterior and dorsal-ventral axes. There are $d^m p^m p^{2p(m-p)}$ possible GRNs with p proteins and m edges (with d edge types). In all search studies, the GRN space is constrained to be connected, have at least one cycle, and with each protein having at least one *in edge*. A GRN is solved by first setting the protein values of each cell from a uniform random distribution, then numerically solving the differential equations using the Runge-Kutta method (with $dt = 0.05$) until either a fixed point (where the average update error $\leq 10^{-8}$ per cell; see Fig.2), or an instability is heuristically detected.

2.3 Determining GRN Computational Adequacy

Our goal is to characterize the space of biological GRNs and determine their computational adequacy to solve spatial design problems found in nature. Three

searches of the space of GRNs were performed. First, an open-ended Monte Carlo search [13] identifies simple GRNs that form interesting or useful spatial patterns. Second, a randomized hill-climbing search identifies simple GRNs that solve the specific spatial patterns of Fig.1. Finally, we searched for combinatorial codes that solve the difficult target patterns of Fig.1 by running multiple disjoint GRNs in parallel.

When searching the space of GRNs we attempt to identify the simplest GRN by systematically considering GRNs of increasing number of proteins and connectivity. In addition, we consider the adequacy of spatial signaling mechanisms by limiting signaling to diffusion or just cell-cell contact, or both. We define diffusion only signaling as ∇, where edges are drawn from equations $A - I$ of Table 1, and contact only signaling as $\diamondsuit$, with edges from equations $C - P$ of the same table.

3 Methods and Results

Monte Carlo Search: The Monte Carlo search samples the GRN space by constructing a GRN as a random graph. We vary the number of proteins $p, 2 \leq p \leq 5$ and the number of edges $m, p \leq m \leq p + 3$. Each GRN is evaluated for its *interestingness* by first solving the GRN with an initial random protein expression then evaluating the spatial regularity of the resulting expression pattern. The algorithm is provided in [27]. The spatial regularity is evaluated by a heuristic measure of interestingness [14] over the frequency terms of the 2D FFT of the protein expression pattern.

Monte Carlo Results: Fig.3 shows some high scoring GRNs and their patterns of expression. Results suggest that the space of GRNs is indeed dense, with weak

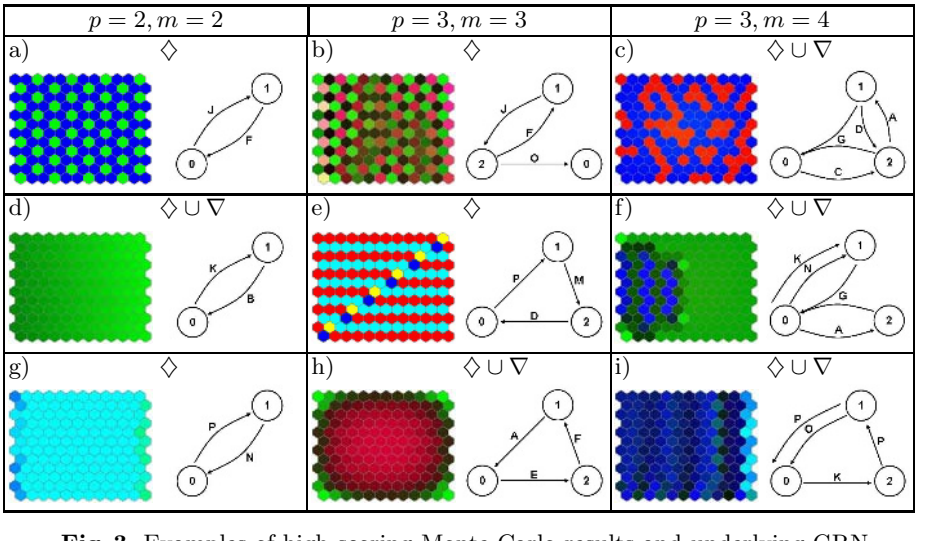


Fig. 3. Examples of high scoring Monte Carlo results and underlying GRN

random sampling identifying many useful GRN's. Small GRN's were found that identify N,S,E,W borders, corners, diagonal partitions, alternating stripes of period 2, 3, 4, and 6, both vertically and horizontally, and mosaics of varying periodicity and regularity. Most interesting were the irregular patterns found but not included in Fig.1, such as mosaic patterned patches (e.g., Fig.3f) and irregular cell clusters (e.g., Fig.3c)

Regular mosaics, patches and borders were all found when edges were restricted to direct contact signaling, demonstrating that this short-range, local form of signaling is a powerful mechanism. Borders and simple segments were found when edges were restricted to diffusible signals, while the irregular patterns on the right of Fig.3 required both direct contact and diffusible signaling.

Just as interesting were the patterns not found. No complex orthogonally segmented patterns such as the checkerboard pattern of *Drosophila* proneural clusters [22] were found (*e.g.* Fig.1k). Neither were complex orthogonal mosaic patterns, seen in the zebra fish retina [21] (*e.g.* Fig.1h). These results suggest that some pattern problems may be too difficult to solve using a single simple ($p \leq 5$, $m \leq 8$) GRN starting from a random distribution.

Hill Climbing Search: The hill climbing search takes target patterns from those given in Fig.1 and attempts to find simple GRN's that best solve each problem. The algorithm is provided in [27]. The mutation operator adjusts the strength parameters ω_k of each edge k by a small random amount. At a lower frequency, the edge *type* (Column 1 Table 1), *in-protein* or *out-protein* are randomly modified. Mutation preserves the connectedness constraint and signaling constraints of the GRN as described above.

To evaluate the fitness of a GRN, its fixed point expression distribution is matched against the provided target pattern. In natural systems, combinatorial codes of protein expression are used to discriminate cell types. The matching function is designed so that combinatorial protein expression codes are discovered, then evaluated as to how they identify cells of the same type, while discriminating cells of different types. Let $D(\sigma, \rho)$ be the Euclidian distance between the protein values of two cells σ and ρ. First, the cells are *k-means* clustered using $D(\sigma, \rho)$ into bins, then the bins are matched against the types assigned in the given target pattern. Cells clustered into the same bin are assigned a unique identifier, $B(\sigma)$. Good clusterings are identified that minimize the distance $D(\sigma, \rho)$ when $B(\sigma) = B(\rho)$ and maximize the distance when $B(\sigma) \neq B(\rho)$. The match is measured by comparing $B(\sigma)$ with the assigned types in the target pattern $T(\sigma)$ (as in [13]) corresponding to distinct colors in Fig.1. To evaluate robustness, each GRN is run over multiple initial random protein distributions and over two cell array sizes (12×12 and 24×24.)

Hill Climbing Results: Fig.4 shows a selection of results executing hill climbing over the patterns in Fig.1. In Fig.4, combinatorial codes are denoted as Boolean expressions over the proteins of cell $P_i(\sigma)$, where H denotes a high expression, L denotes a low expression, and M denotes an intermediate expression corresponding to cluster values found by *k-means*. As suggested by the Monte Carlo stud-

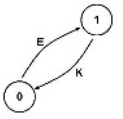
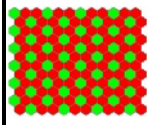
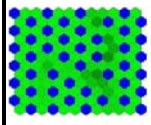
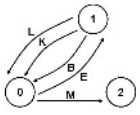
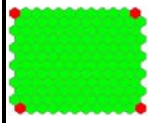
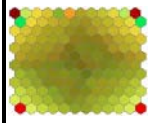
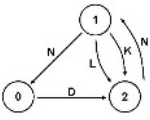
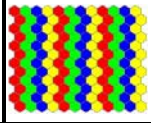
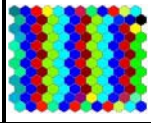
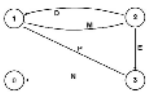
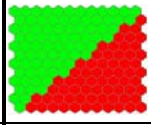
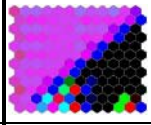
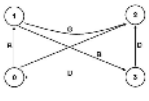
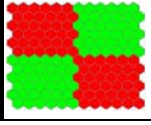
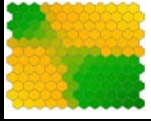
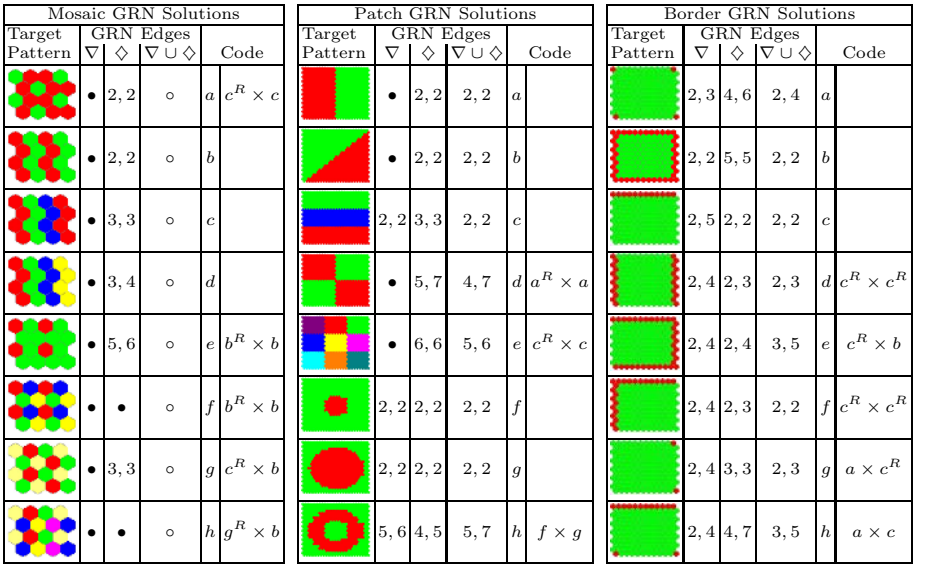
	Signaling	GRN Graph	Code	Target	Solution
a)	♢		$P_0(\sigma)$ L H $P_1(\sigma)$ H L $B(\sigma)$ 0 1		
b)	♢ ∪ ∇		$P_0(\sigma)$ H L $P_1(\sigma)$ L H $P_2(\sigma)$ L H $B(\sigma)$ 0 1		
c)	♢		$P_0(\sigma)$ L L H L $P_1(\sigma)$ M L L H $P_2(\sigma)$ H H L L $B(\sigma)$ 0 1 2 3		
d)	♢		$P_0(\sigma)$ L H $P_1(\sigma)$ L L $P_2(\sigma)$ L H $P_3(\sigma)$ H L $B(\sigma)$ 0 1		
e)	♢ ∪ ∇		$P_0(\sigma)$ L L $P_1(\sigma)$ H H $P_2(\sigma)$ H L $P_3(\sigma)$ L L $B(\sigma)$ 0 1		

Fig. 4. Examples of high scoring Hill Climbing runs. Each row is the best run found for the target pattern. The signaling constraint, simplest GRN, the combinatorial code discovered, the target pattern and expression pattern are shown in each row.

ies, many target patterns were easily solved from initially random distributions. For example, the GRN and combinatorial code shown in Fig.4a is the *dual* of the classic solution for lateral inhibition with feedback for mosaic construction [20]. Significantly, like the Monte Carlo simulation, simple hill climbing from an initially random pattern failed to create many biologically relevant patterns, including complex orthogonal mosaics (*e.g.* Fig.1h) and patches (*e.g.* Fig.1k).

Table 5 summarizes a systematic computational exploration of adequacy, where the simplest GRN was identified using only direct contact signaling ♢, or only diffusible signaling ∇, or both signaling types. Target patterns were selected from each problem class (mosaic, patch and border).

Mosaics: Diffusion signaling alone is inadequate to create mosaics, while direct contact signaling is both necessary and sufficient. Furthermore, GRNs for complex mosaics, such as those in the zebra fish retina [21] (*e.g.* Fig.1h, where h denotes the Code) are difficult to find when searching from random GRNs. This suggests an alternative incremental evolutionary strategy of mutating GRNs that solve related mosaics. An example of related GRNs is shown in Fig.3. The GRN in Fig.3b that solves this complex pattern is a mutant version of the simpler GRN in Fig.3a, the difference being the addition of one protein and one edge. We are currently exploring the power of this evolutionary strategy.



Mosaic GRN Solutions						Patch GRN Solutions						Border GRN Solutions					
Target Pattern	GRN Edges ▽	◇	▽ ∪ ◇		Code	Target Pattern	GRN Edges ▽	◇	▽ ∪ ◇		Code	Target Pattern	GRN Edges ▽	◇	▽ ∪ ◇		Code
	•	2, 2	○	a	$c^R \times c$		•	2, 2	2, 2	a			2, 3	4, 6	2, 4	a	
	•	2, 2	○	b			•	2, 2	2, 2	b			2, 2	5, 5	2, 2	b	
	•	3, 3	○	c			2, 2	3, 3	2, 2	c			2, 5	2, 2	2, 2	c	
	•	3, 4	○	d			•	5, 7	4, 7	d	$a^R \times a$		2, 4	2, 3	2, 3	d	$c^R \times c^R$
	•	5, 6	○	e	$b^R \times b$		•	6, 6	5, 6	e	$c^R \times c$		2, 4	2, 4	3, 5	e	$c^R \times b$
	•	•	○	f	$b^R \times b$		2, 2	2, 2	2, 2	f			2, 4	2, 3	2, 2	f	$c^R \times c^R$
	•	3, 3	○	g	$c^R \times b$		2, 2	2, 2	2, 2	g			2, 4	3, 3	2, 3	g	$a \times c^R$
	•	•	○	h	$g^R \times b$		5, 6	4, 5	5, 7	h	$f \times g$		2, 4	4, 7	3, 5	h	$a \times c$

Fig. 5. The computational adequacy of GRN's for solving selected mosaic, patch and border target patterns. The *GRN Edges* column displays the solution found under different signaling constraints: contact only ◇; diffusion only ▽; or their union ◇ ∪ ▽. GRN solutions are noted: •- no solution found; p, m solution found (p proteins and m edges); ○- poor solutions found. The *Code* column gives the disjoint code solution found, where $a \times c^R$ concatenates solution a and the solution of rotated GRN c. A GRN is rotated by replacing any directional signaling edges M:P (in Table 1) with the signaling edge from the rotated direction (such as north goes to west, and west goes to south etc.).

Patches: Diffusible signals are effective for finding circular (Fig. 5g,h) and 1/3 patterns (Fig. 5c), but ineffective at finding 1/2 patterns (Fig. 5 a,b). Significantly, short-range direct contact signaling is effective at building global structures like segments, including 1/2, 1/3 and nested circles (Fig. 5a-h). The emergence of these global patterns as an outcome of short-range signaling is a significant result of this work. Finally, GRNs that combine diffusion and direct contact signaling are effective and parsimonious, and can solve all given segment problems.

Borders: Border targets appear to be the simplest to solve, with both diffusion and cell-cell contact signaling adequate and effective. In some cases combining both diffusion and contact signaling creates the most parsimonious GRNs.

Combining GRNs Search: In the final study we identify combinatorial codes that solve the difficult targets of Fig.1 by running multiple disjoint GRNs in parallel. Only GRNs identified by hill climbing are considered. To run two disjoint GRNs in parallel the GRNs are assigned distinct proteins then both are run until fixed point. Next their output protein distributions are concatenated, then the combined expression patterns are clustered and matched against the target pattern as described in Section 3.

Combining GRNs Results: The results of running two GRNs in parallel are shown for each target pattern in the last column of Fig.5. Complex orthogonal target patterns (*e.g.* Fig.1k), which were hard to solve using a single GRN, can easily be solved by this method. For example, a single GRN of 5 proteins and 7 edges (both contact and diffusion) is needed to identify the nested circular pattern (Fig.5h), whereas two small (2 protein, 2 edge) GRNs can solve it easily.

4 Conclusions

In this work we show that simple biologically realistic GRNs are very powerful and capable of constructing many complex cellular patterns. One significant and unanticipated result is that cell-cell contact signaling is sufficient to form many global patch patterns. The complexity of the cellular patterns formed by the simple GRNs modeled here poses the question of why biological GRNs are so complex. This study revealed some patterns that could not be solved by a single small GRN. These patterns, however, were solvable when disjoint GRNs were run in parallel and their protein expression levels combined. Our results support the view that complex GRNs may have evolved in nature by combining simpler modules. Our model provides an avenue to explore the evolution of complex GRNs that drive development.

Acknowledgements

The authors wish to thank Dharmesh Shar, Ranjitha Dhanasekaran, Swapna Challa and Prashant Rai for help in preparing the paper and writing the code.

References

1. Stanley, K., Mikkulainen, R.: A Taxonomy For Artificial Embryogeny. Artificial Life 9(2): (2003) 93–130
2. Lindenmayer, A.: Mathematical Models For Cellular Interaction In Development: Parts I and II. Journal of Theoretical Biology. **18** (1968) 280–315
3. Gruau, F.: Neural Network Synthesis Using Cellular Encoding And The Genetic Algorithm. Doctoral dissertation, Ecole Normale Superieure de Lyon, France (1994)
4. Gruau, F., Whitley, D., & Pyeatt, L.: A Comparison Between Cellular Encoding And Direct Encoding For Genetic Neural Networks. In J. R. Koza, D.E. Goldberg, D. B. Fogel, & R. L. Riolo (Eds.), Genetic Programming 1996: Proceedings of the First Annual Conference, Cambridge, MA: MIT Press (1996) 81–89
5. Hogeweg, P. Computing an organism: on the interface between informatic and dynamic processes. Biosystems, (2002) 64: 97-109
6. Astor J.S. & Chris Adami C. A Developmental Model for the Evolution of Artificial Neural Networks. Journal of Artificial Life. (2000) **6:3** 189-218
7. Turing, A.: The Chemical Basis Of Morphogenesis. Philosophical Transactions of the Royal Society B. **237** (1952) 37–72
8. Fleischer, K., Barr, A.H.: A Simulation Testbed For The Study Of Multicellular Development: The Multiple Mechanisms Of Morphogenesis. In C. G. Langton (Ed.), Artificial life III. Reading, MA. Addison-Wesley (1993) 389–416

9. Mjolsness, E., Sharp, D. H., & Reinitz, J.: A Connectionist Model Of Development. Journal of Theoretical Biology. **152** (1991) 429–453
10. Kaneko, K., Furusawa, C.: Emergence Of Multicellular Organisms With Dynamic Differentation And Spatial Pattern. Artificial Life. **4** (1998) 77–93
11. Federici, D.: Using Embryonic Stages To Increase The Evolvability Of Development. In proceedings of WORLDS Workshop on Regeneration and Learning in Developmental Systems, hosted by GECCO (2004)
12. Roggen, D., Federici, D.: Multi-Cellular Development: Is There Scalability And Robustness To Gain? In proceedings of PPSN VIII 2004 The 8th International Conference on Parallel Problem Solving from Nature (2004) 391-400
13. von Dassow, G., Meir, E., Munro, E.M., & Odell, G.M.: The Segment Polarity Network is a Robust Developmental Module. Nature. **406** (2000) 188–192
14. Hilderman, R.J., Hamilton, H.J.: Heuristics For Ranking the Interestingness Of Discovered Knowledge. In N. Zhong and L. Zhou, editors, Proceedings of the Third Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'99), Beijing, China, (1999) 204–209
15. Izaguirre, J.A., Chaturvedi, R., Huang, C., Cickovski, T., Coffland, J., Thomas, G., Forgacs, G., Alber, M., Hentschel, G., Newman, S.A., & Glazier J,A.: CompuCell, a Multi-Model Framework For Simulation Of Morphogenesis. Bioinformatics. **20** (2004) 1129-37
16. Bongard, J. C.: Evolving Modular Genetic Regulatory Networks. In Proceedings of the IEEE 2002 Congress on Evolutionary Computation (CEC2002), IEEE Press, (2002) 1872–1877
17. Taylor,T.: A Genetic Regulatory Network-Inspired Real-Time Controller For a Group Of Underwater Robots. In Intelligent Autonomous Systems 8 (Proceedings of IAS8), F. Groen, N. Amato, A. Bonarini, E. Yoshida and B. Krse (eds.), IOS Press, Amsterdam, (2004) 403–412.
18. Lawrence, P.A.: The Making Of a Fly. Blackwell Scientific Publications, Oxford (1992)
19. Shvartsman, S.Y., Muratov, C.B., & Lauffenburger, D.A.: Modeling And Computational Analysis Of EGF Receptor-Mediated Cell Communication In Drosophila Oogenesis. **129** Development (2002) 2577–2589
20. Collier, J.R., Monk, N.A.M., Maini, P.K., & Lewis, J.H.: Pattern Formation By Lateral Inhibition With Feedback: A Mathematical model Of Delta-Notch Intercellular Signalling. **183** J. Theor. Biol. (1996) 429–446
21. Mochizuki,A.: Pattern Formation Of the Cone Mosaic In the Zebrafish Retina: A Cell Rearrangement Model. J. Theor. Biol. **215** (2002) 345–361
22. Skeath, J.B.: At the Nexus Between Pattern Formation And Cell-Type Specification: the Generation Of Individual Neuroblast Fates In the Drosophila Embryonic Central Nervous System. Bioessays. **212** (1999) 922–931
23. Jessell, T.M.: Neuronal Specification In the Spinal Cord: Inductive Signals And Transcriptional Codes. Nat. Rev. Genet. 1 (2000) 20–29
24. Cook, J.E. and Chalupa, L.M.: Retinal Mosaics: New Insights Into An Old Concept. TINS **23** (2000) 26–35
25. Atabef, M., Clarke, J.D.W., Ticle, C.: Dorso-Ventral Ectodermal Compartments and the Origin of Apical Ectodermal Ridge in Developing Chick Limb. Development **124** (1997) 4547–4556
26. Gossler, A., de Angelis, M.H.: Somitogenesis. Curr. Top. Dev. Biol. **38** (1998) 225–287
27. `http://www.cs.usu.edu/~flann/index.html`

A Coarse-Coding Framework for a Gene-Regulatory-Based Artificial Neural Tissue

Jekanthan Thangavelautham and Gabriele M.T. D'Eleuterio

Institute for Aerospace Studies, University of Toronto,
Toronto, Ontario, Canada, M3H 5T6
thangav@ecf.utoronto.ca, gabriele.deleuterio@utoronto.ca

Abstract. A developmental Artificial Neural Tissue (ANT) architecture inspired by the mammalian visual cortex is presented. It is shown that with the effective use of gene regulation that large phenotypes in the form of Artificial Neural Tissues do not necessarily pose an impediment to evolution. ANT includes a Gene Regulatory Network that controls cell growth/death and activation/inhibition of the tissue based on a coarse-coding framework. This scalable architecture can facilitate emergent (self-organized) task decomposition and require limited task specific information compared with fixed topologies. Only a global fitness function (without biasing a particular task decomposition strategy) is specified and self-organized task decomposition is achieved through a process of gene regulation, competitive coevolution, cooperation and specialization.

1 Introduction

Evolving open-ended variable-length neural systems with large phenotypes remains a significant challenge in the field of Alife [20,8]. One of the problems encountered with large phenotypes is the bootstrap problem [18]. The bootstrap problem occurs when the EAs are unable to pick out incrementally better solutions for crossover and mutation resulting in premature stagnation of the evolutionary run. The answer to the evolution of controllers for complex problems has often been to introduce more supervision *ad hoc*, where the experimenter decomposes a complex task into a set of simpler tasks based on domain knowledge of the task at hand. In biological systems, such intervention (more supervision) does not always exist, yet these systems can adapt and thrive with relative ease.

Other techniques involve starting with a single cell or a small phenotype and allowing for the system to grow in size and complexity until the system can find a satisfactory solution to a given task [14]. However in biological systems, often there exists a brain that may already have the neural capacity (brain size) to adapt easily to a new task/scenario. In such circumstances, starting over with a minimalist system may be a much slower process owing to the over-reliance of topological growth directed by evolution.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 67–77, 2005.

Taking inspiration from the mammalian visual cortex, we have developed an evolvable Artificial Neural Tissue (ANT) model. The genotype for the evolvable ANT defines a developmental program that constructs a neural tissue (phenotype) and associated gene-regulatory functionality. The variable-length tissue architecture can be characterized as a lattice of neurons arranged in a three-dimensional (3-D) structure. A Gene Regulatory Network (GRN) controls cell division and cell death in the tissue, activates/inhibits portions of the tissue based on external sensory input using a coarse-coding framework and express/repress other characteristics based on gene-protein interactions.

We empirically compare the training performance (using evolutionary algorithms) of various controllers for the multirobot tiling pattern formation task (similar to a segment of the termite nest building task [4]),variant of the phototactic task (with obstacles and robot equipped with a gripper) and a relatively difficult sign-following task that requires use of memory. Each of these tasks requires the evolution of emergent (self-organized) task decomposition strategies to complete the task successfully. Only a global fitness function (without biasing a particular task decomposition strategy) is used; the intention is for the controllers to evolve innovative techniques in decomposing the global task into a set of 'local' subtasks.

2 Related Work

Traditional machine learning methods for task decomposition involve the supervisor decomposing the task and training separate 'expert networks' on the subtasks [12]. Arbitration among the expert networks is performed using a cooperative (Product of Experts model) [10] or competitive Mixture of Experts model [12,17]. The gating function and the expert networks are trained separately and the network topology is predetermined by the experimenter.

Our previous work took this approach to the next step, where decision neurons (acting like gating functions) and expert networks were evolved together (Binary Relative Lookup architecture) using a global fitness function [21]. We found larger BRL architectures (with more expert networks) tend to evolve faster than comparable smaller ones for the tiling pattern formation task. The decision neurons learned to limit the number of expert networks used thus preventing problems in over segmentation (over-fitting) to many expert networks.

NEAT (NeuroEvolution of Augmenting Topologies) showed the potential advantage of evolving both the neuron weights and topology concurrently [13]. It is argued that growing the network incrementally ('complexification') serves to minimize the dimensional search space and thus improve evolutionary performance. ANT is even more flexible and can be initialized with a large number of neurons since the GRNs can effectively suppresses unnecessary/noisy neurons while activating neurons specialized for specific input signals.

Another approach to evolving solutions to complex tasks involves use of encoding schemes that effectively reduces the search space. This includes a multicellular developmental system by Eggenberger [5] and the Morphogenetic Sys-

tem (MS) originally used on POEtic [19]. Eggenberger's earlier model demonstrates how cell differentiation, cell division and cell death can be controlled by gene regulatory functionality and constructs a 3-D organism. In these two systems, the GRNs merely act on the developmental process in constructing the phenotype.

A more refined model by Gomez and Eggenberger [6] uses 'ligand-receptor interactions' allowing for one neuron to recognize/attach to a partner neuron and allow for emergence of Hebbian-type learning without specification of learning rules for a forveating artificial retina system. Astor and Adami's [2] tissue architecture consists of cells on a 2D tissue that perform logical functions. Cell replication and connections are formed through a gene regulated developmental and learning system using a Genetic Programming type command set.

In our ANT architecture, gene regulation occurs during the developmental process in addition to when the tissue interacts with the environment. The decision protein act much like gating neurons while helping to reduce the effects of *spatial crosstalk* [12] and perform sensory preprocessing enabling selection of 'specialized' networks of neurons depending on the sensory input.

Another class of indirect developmental encoding schemes such as Artificial Embrogeny systems [14] produce phenotypes through recursive rewriting of the genotype code. These systems use an artificial chemistry as a grammar or model cellular metabolism and replication. Other recursive rewriting schemes include Cellular Encoding [7] and L-Systems (see [15,11]).

It has been argued that indirect developmental encoding schemes may effectively decrease the search space (by exploiting regularities and allowing for peleiotropy) but at the price of introducing a deceptive fitness landscape [20]. It has also been found the overhead required for indirect encoding schemes appear to result in poor performance for smaller search spaces [20]. This presents a problem for task decomposition, where the control scheme needs to work well for subtasks with small and large search spaces.

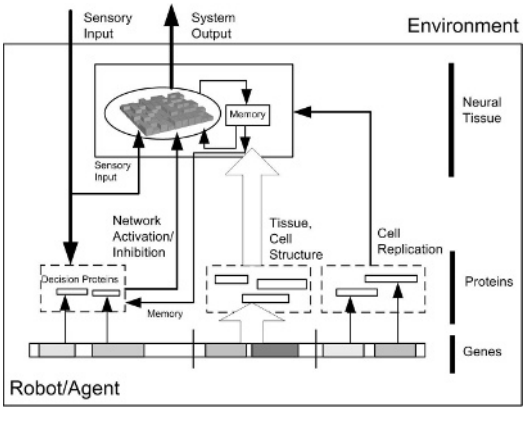


Fig. 1. Schematic of the ANT architecture showing gene-protein-tissue interaction

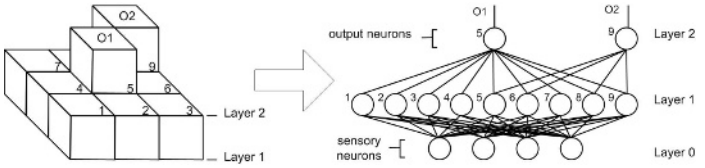


Fig. 2. Diagram of tissue morphology and equivalent networks where O1 and O2 are output signals

3 Methodology

The ANT architecture presented in this paper consists of a developmental program that constructs a neural tissue and associated gene-regulatory functionality. The gene-regulatory network consists of parameters that control growth and activates/inhibits parts of the genotype. All the parameters defined within the ANT architecture are evolved. This includes parameters characterizing the decision proteins, growth parameters, cell contents and tissue topology. Neural networks consisting of cells within the tissue are dynamically formed through a sequence of activation/inhibition based on the coarse-coding framework.

The artificial tissue consists of a culture of cells activated and inhibited by a gene-regulatory network. The cells exist in a three-dimensional matrix with each cell occupying a cube (Fig. 2). Each cell contains genes specifying weights, thresholds/biases, choice of activation function (modular neuron model [21]) and probability ratios for instructing cell division. Cell division requires a parent cell (selected with highest replication probability using GRNs). The new cell can be located in one of 6 neighbouring locations (top, bottom, north,south,east,west) sharing a common side with the parent and is not occupied by another cell.

The base layer of cells within the tissue are fully connected to all the sensory neurons or memory neurons. While the top layer of cells triggers a set of predefined basis behaviours such as to perform motor control or pass data to memory neurons. Connection between cells from one layer to another is local, with each cell from layer z connected to a maximum of $m = 9$ cells from layer $z-1$ (spatially localized). $p_{x,y,z} = (\sum_{i=x-1}^{x+1}(\sum_{j=y-1}^{y+1} w_{i,j,z-1}s_{i,j,z-1}))/(\sum_{i=x-1}^{x+1}\sum_{j=y-1}^{y+1} s_{i,j,z-1})$ and $s_{x,y,z} = [\phi_n(p,t_1,t_2)]_{x,y,z}$ where $w_{x,y,z}$ is a neuron weight and $s_{x,y,z}$ is the current state of a neuron. The modular neuron model used consists of two threshold parameters t_1 and t_2, where each neuron outputs one of two states $s = (s_1, s_2)$. A choice of four threshold activation functions for ϕ_n is given below:

$$\begin{array}{ll} \phi_1: & s_{out} = \begin{cases} s_1, & \text{if } p \le t_1 \\ s_2, & \text{if } p > t_1 \end{cases} \\ \phi_2: & s_{out} = \begin{cases} s_1, & \text{if } p \ge t_2 \\ s_2, & \text{if } p < t_2 \end{cases} \end{array} \quad \begin{array}{ll} \phi_3: & s_{out} = \begin{cases} s_1, & \text{if } t_2 < p < t_1 \\ s_2, & \text{otherwise} \end{cases} \\ \phi_4: & s_{out} = \begin{cases} s_1, & \text{if } p > (1-p) \\ \text{rand}(s_1,s_2), & \text{if } p = (1-p) \\ s_2, & \text{if } p < (1-p) \end{cases} \end{array} \tag{1}$$

enabling a single neuron to simulate AND, OR, NOT and XOR functions.

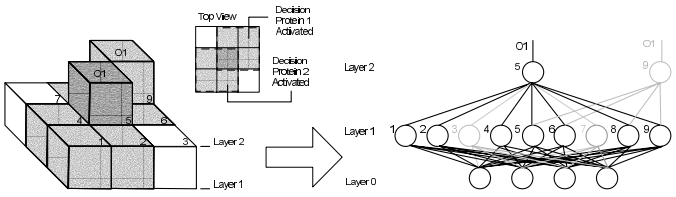


Fig. 3. Diagram showing network with neuron 5 (layer 2) with highest activation concentration (due to coarse activation of decision protein 1 and 2) being selected

3.1 Decision Proteins and Coarse Coding

Albus argued that the mammalian brain uses tile (coarse) coding to represent multidimensional space [1,9]. Hinton [9] and Ballard [3] point to the importance of modularity in the coarse-coding framework as an increase in the number of neurons in a fixed volume limits the number of dimensions represented. We show in our model, the potential advantages of coarse-coding as a means of arbitration between modular networks.

The activation and inhibition of genes occur through a coarse-coding framework. Decision proteins (modeled as single neurons with ability to choose between threshold activation functions) get activated and inhibited due to sensory input. The decision proteins act by diffusing through a coarse column (receptive field) as shown in Fig. 3. The receptive field parameters specifying position and dimensions ($D_i[x, y, x_{length}, y_{length}]$) for each decision protein D_i is also evolved.

Once a decision protein is activated, the activation concentration $C_i[c_{active}]$ of each neuron cell i is incremented by k (a constant). With multiple decision proteins acting together, a consensus is reached when the activation concentration (summed over multiple activated proteins) is highest for a particular output neuron (cell at the top layer of a column). A network is dynamically formed from all the neurons connected to the selected output neuron (in a scheme shown in Fig. 3) with activation concentration ($c_{active} > 0$). This characteristic is inspired from the columnar pooling of neurons within the mammalian visual cortex. Similarly, if there are multiple output neurons with the same activation concentration, selection occurs among the output neurons according to a uniform distribution.

3.2 Introns

The accumulation of nonsense genes or introns has been described to be a major problem in the field of Genetic Programming. However, introns allow for genetic neutrality, an important facet of evolution. With current GP approaches, intron accumulation results in increased computational inefficiency over time. Current techniques in controlling introns has been to prune the genotype regularly or explicitly impose a size limit on the genotype using the fitness function. Both strategies either lack biological plausibility or translate into more supervision.

It is noted that with GP approaches, program size increases according to the square power law in generations [16]. For ANT, the program size growth is

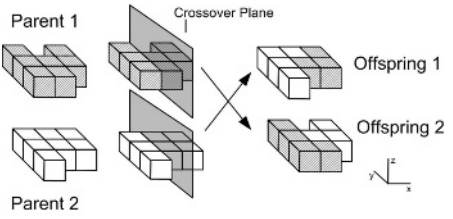


Fig. 4. A crossover operation between two ANT parents. 'Compatible' cells are interchanged as shown resulting in two offspring

controllable and increases linearly due to the growth model used. Our genotype-to-phenotype mapping scheme avoids intron accumulation due to the crossover operator. In GP, a node is chosen from each genome and genetic code is exchanged about the node. In our methodology , only a 'compatible set' of genes get interchanged during crossover. Each cell has a unique position parameter $[x, y, z]$ relative to rest of the cells within the tissue. A crossover is performed by drawing a plane (with normal vector parallel to the x or y-axis) separating the tissue and exchanging 'compatible' cells (Fig. 4). Thus genes for cell C_1 from Tissue A and C_2 from Tissue B could be interchanged iff $C_{A,1}[x, y, z] = C_{B,2}[x, y, z]$.

4 Example Tasks

The evolutionary performance of our ANT architecture is compared with fixed network topologies (direct-encoding schemes) for three different robotic tasks. All three robotic tasks were chosen because it could be argued that self-organized task decomposition strategies may be necessary to accomplish the tasks given a 'global' fitness function. In addition, these tasks are inspired by some remarkable behaviours evident in the insect world. The tasks include a multirobot tiling pattern formation task [21] that involves redistributing objects (blocks) randomly placed in a two-dimensional world into a desired tiling structure (see Fig. 7). The robots need to come to a consensus and form one 'perfect' tiling pattern. This task is similar to a segment of the termite nest construction task that involves redistributing pheromone filled pellets on the nest floor [4].

The tiling formation task may be decomposed into a set of subtasks such as foraging for blocks, redistributing block piles, arranging blocks in the desired tiling structure locally, merging local lattice structures, reaching a collective consensus and finding/correcting mistakes. In the phototactic task, the robot must reach a goal location (light source), but it also needs to negotiate obstacles.

In the sign-following task, the robot needs to evolve the ability to decipher the signs relative to the robot's current frame of reference,to remember the current sign while looking for the next one, and negotiate obstacles. Each sign is a waypoint that gives direction to the next waypoint leading ultimately to a goal location. Of the three tasks, this task is the most complex and requires the use of memory. This task is inspired by honey bees ability to waggle (communicate with other bees) and describe directions and waypoints to a food source.

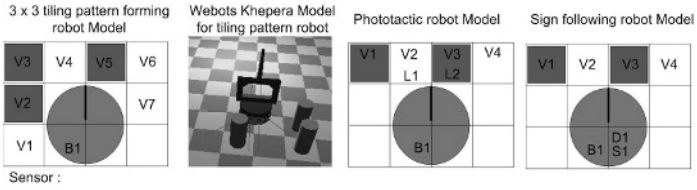


Fig. 5. Schematic of 2D robot model. B1 detects whether robot is carrying a block, D1 is a compass sensor and S1 reads the colour of the sign off the floor.

The signs are posted on a fixed frame of reference and the robot based on its current heading needs to interpret the signs accordingly. For the phototactic task and the sign-following task, the fitness function is simply the number of times the robot reaches and stays at the goal location and for the tiling formation task the fitness is the Shannon's entropy over all the tiles [21].

The robots are modeled as Kheperas equipped with a gripper turret. We have developed a fast two-dimensional grid world simulator for our robotic experiments and we have verified the performance of the evolved controller for the phototactic and tiling pattern formation task using Cyberbotic's Webots (Fig. 5). The basis behaviours chosen for all three tasks are shown in the table below:

4.1 Experiments

The evolutionary performance of various control system architectures is compared for the three tasks (see Fig. 6). The fixed network architectures (Table 1) were determined based on the number of neurons required for triggering each basis behaviour and the size of the networks were limited to avoid the 'bootstrap problem'. BRL2 consist of 2 monolithic networks (Table 1) arbitrated by a decision neuron. ANT is initialized with individuals between 40 to 400 cells (uniform distribution) for the tiling formation task and 20 to 100 cells for the phototactic and sign-following task. This procedure is intended to determine wether there is evolutionary preference for smaller phenotypes over larger ones.

Task	3x3 tiling pattern formation	Phototactic	Sign Following
Basis Behaviours	Move Pickup/Putdown NOP	Pickup/Putdown Move Forward Turn Right Turn Left NOP	Pickup/Putdown Move Forward Turn Right Turn Left NOP
Memory Neurons*		2	4
Monolithic Network Topology	4 hidden neurons, 1 output neuron	12 hidden neurons, 4 output neurons (without memory) 8 output neurons (with memory)	36 hidden neurons, 12 output neurons

Fig. 6. Basis behaviours for the three robotic tasks and monolithic network topology shown. A combination of these behaviours can be activated at each timestep.

The EA population size for all three experiments is $P = 100$, crossover probability $p_c = 0.7$, mutation probability $p_m = 0.025$ and tournament size of $0.06P$ (for tournament selection). For the phototactic task and the sign-following task the success rate is averaged over 100 different initial conditions (consisting of 20×20 world, 80 blocks) and for the tiling formation task, the fitness is averaged over 15 initial conditions. For the sign-following task 'mines' (undetectable by robot) are randomly laid throughout the gridworld except along the pathway.

5 Results and Discussion

The evolutionary performance (number of successful epochs) of the ANT architecture for three different robotic tasks is better than smaller fixed network topologies (Fig. 6). In addition, ANT with its ability to grow in complexity (depending on task) and exploit modularity managed to find solutions to the sign-following task (memory dependent) where fixed topologies appear to fail.

Analyzing the 3-D morphology, active segments (consisting of specialized networks) are distributed sparsely throughout the tissue. These networks do not appear to decompose the task according to 'recognizable' distilled behaviours but as a set of emergent proximal behaviours (proximity in sensor space) [18].

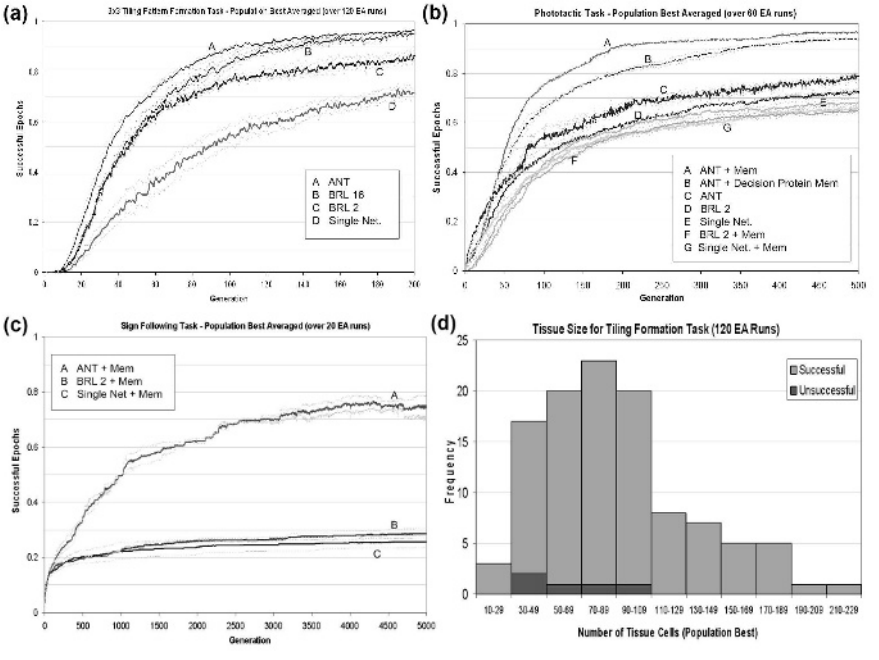


Fig. 7. Evolutionary performance comparison for the tiling-formation (a), phototactic (b) and sign-following tasks(c). Also shown, a histogram (d) of number of cells within the tissue (population best, after 200 generations), for the tiling formation task.

Large tissue structures do not seem detrimental to the evolutionary performance because the GRN quickly evolves the ability to suppress unnecessary/noisy neurons within the tissue. There appears to be a steady reduction in the number of active neurons with a convergence to a solution (Fig. 7b). During the early phase of evolution, there is greater network activity enabling sampling of more neurons. Convergence appears to result in the inhibition of cells that add noise to the system (reduction in spatial crosstalk [12]).

There appears to be a noticeable improvement in evolutionary performance of large tissue structures over smaller ones. A histogram of population best during successful runs (with a success rate of > 0.9) compared to population best during unsuccessful runs (success rate < 0.1) appear to confirm these observations (Fig. 6d). The ANT architecture is more effective at being able to decompose a task and handle the subtasks using simpler specialized networks consisting of few neurons than the fixed architectures. This results in fewer weights having to be tuned and thus a smaller search space.

The most promising result from our experiments is the successful exploitation of global memory by the ANT architecture. While a fully connected feed forward network and the BRL architecture were unable to exploit this functionality resulting in lower performance (due to a larger search space). This observation corroborates Nolfi's experiments for the garbage collection task [18].

The ANT architecture consists of cells interconnected locally but with access to global memory, specialized networks separated by distance have a means of communication, resulting in output signals that are maybe less contradictory. Access to global memory allows for some emergent (self-organization) properties to appear with one segment of the tissue able to 'veto' or override another segment of the tissue independently. This is evident from significant performance improvement for the phototactic task when only the decision proteins were able to read the memory neurons (Fig. 6b).

It is also observed that the number of active decision proteins steadily increases even after convergence to a solution thus creating a redundant protective mechanism against adverse mutation (Fig. 7b). Redundancy is an important

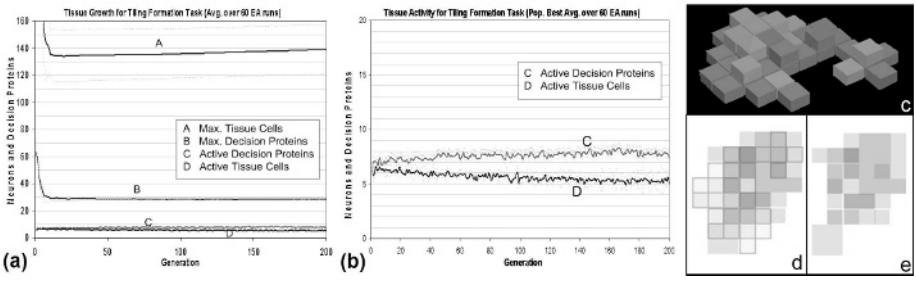


Fig. 8. Tissue growth characteristics (a) and max number of active cells/decision proteins (b). (c) 3D morphology of a tissue (solution for tiling formation task). Top view (d) showing all the decision proteins/cells and top view (e) of decision proteins.

characteristic of biological systems that enable a system as whole to be robust despite the fragility of its parts. From an evolutionary standpoint, the artificial tissues evolve not only to solve the intended task, but also steadily develop ways to ensure its survival and successful transfer of its genes.

6 Conclusion and Future Work

A developmental Artificial Neural Tissue architecture, with the ability to perform emergent task decomposition is presented in this paper. The evolutionary performance of the ANT architecture is better than smaller fixed network topologies for three different robotic tasks. Our experiments indicate improved evolutionary performance of ANT with access to memory neurons. It is hypothesized that memory neurons allow for communication among the specialized networks, enabling one network to override another. We are currently planning to port ANT onto hardware and hope to compare our architecture with other variable length topologies for tasks such as soccer, mining and surface exploration.

Acknowledgements. The authors would like to thank NSERC and the Ontario Government for supporting this research and to the reviewers for their helpful comments.

References

1. Albus, J. S. A theory of cerebellar function. Mathematical Biosciences, (1971) 25–61.
2. Astor JC, Adami C. 2000. A developmental model for the evolution of artificial neural networks. ALife 6(3):189-218
3. Ballard, D.H., Cortical Connections and parallel processing. The Behavioural and Brain Sciences, Vol. 9, (1986) 279–284
4. Deneubourg J-L. Application de l'ordre par fluctuations 'a la description de certaines 'etapes de la construction du nid chez les termites. Insectes Sociaux, (1977) 117–130
5. Eggenberger, P., et al., Evolving the orphology of a neural net for controlling a foveating retina, ALife 8 (2002) 243-251
6. Eggenberger, P., Evolving Morphologies of simulated 3d organism based on Differential Gene Expression, Proc. 4th European Conference on ALife (1997)
7. Gruau, F. Automatic definition of modular neural networks. Adaptive Behaviours 3 (1994) 151–183
8. Gustafson S., et al.,Problem Difficulty and Code Growth in Genetic Programming Genetic Programming and Evolvable Machines 5, (2004) 271–290
9. Hinton, G., Shape Representation in Parallel Systems, Proc. of 7th Int. Joint Conf. on Artificial Intelligence (1981) 1088–1096
10. Hinton, G.,Product of Experts, Proc. of the 9th Int. Conf. on Artificial Neural Nets (1999) 1:1-6
11. Hornby, G., Pollack, J. Creating High-Level Components with a Generative Representation for Body-Brain Evolution. ALife 8, 2002

12. Jacobs R., Jordan M., Barto A., Task decomposition through competition in a modular connectionist architecture. Cognitive Science, No. 15, (1991) 219-250
13. Stanley K.,Miikkulainen R.,Continual Coevolution through Complexification in Proc. of the Genetic and Evolutionary Computation Conf. 2002, (2002)
14. Stanley K.,Miikkulainen R., A taxanomy for aritificial embrogeny., Artificial Life 9, (2003) 2:93-130
15. Sims K., Evolving 3D Morphology and Behavior by Competition, Proc. of Artificial Life IV, (1994) MIT Press, 28–39.
16. Langdon, W., Quadratic bloat in genetic programming, in Proc. of Genetic and Evolutionary Comp. Conf. (2000)
17. Liu Y.,Yao X., Higuchi T., Evolutionary Ensembles with Negative Correlation Learning. IEEE Transactions on Evolutionary Computation (2000) 4:380-387
18. Nolfi, S., Floreano D.: Evolutionary Robotics : The Biology, Intelligence, and Technology of Self-Organizing Machines, MIT Press (2000) 13-15
19. Roggen D., Floreano, D., Mattiussi, C., A Morphogenetic Evolutionary System: Phylogenesis of the POEtic Tissue, Int. Conf on Evolvable Systems (2003) 153–164
20. Roggen D., Federici D., Multi-cellular Development: Is There Scalability and Robustnes to Gain?, In Proc. of Parallel Problem Solving from Nature (2004) 391–400
21. Thangavelautham, J.,D'Eleuterio, G.M.T, A Neuroevolutionary Approach to Emergent Task Decomposition, Proc. of 8th Parallel Problem Solving from Nature (2004) 991-1000

A Computational Model of Cellular Morphogenesis in Plants

Tim Rudge[1] and Jim Haseloff

Department of Plant Sciences, University of Cambridge, Cambridge CB2 3EA, U.K
[1] Present address: ARC Centre for Complex Systems and Advanced Computational Modelling Centre, University of Queensland, QLD 4072, Australia
timrudge@maths.uq.edu.au
jim.haseloff@plantsci.cam.ac.uk

Abstract. Plant morphogenesis is the development of plant form and structure by coordinated cell division and growth. We present a dynamic computational model of plant morphogenesis at cellular level. The model is based on a self-reproducing cell, which has dynamic state parameters and spatial boundary geometry. Cell-cell signalling is simulated by diffusion of morphogens, and genetic regulation by a program or script. Each cell runs an identical script, equivalent to the genome. The model provides a platform to explore coupled interactions between genetic regulation, spatio-mechanical factors, and signal transduction in multicellular organisation. We demonstrate the capacity of the model to capture the key aspects of plant morphogenesis.

1 Introduction

Plant morphogenesis is the formation of shape and structure by coordination of cell shape, growth, and proliferation by mitosis. The control mechanisms involved in regulating morphogenesis are complex, and reverse engineering them from experimental data is an extremely difficult task. Thus, computational and mathematical models are becoming increasingly important tools for developmental biologists.

1.1 Plant Morphogenesis

Plant cells are enclosed in a semi-rigid cell wall composed of cellulose microfibrils, other polysaccharides and proteins[1]. They secrete an extracellular matrix which binds the walls of neighbouring cells into fixed relationships - each cell is firmly bound to its neighbours[1]. The partition between two cells is *double-walled*, with each cell's wall having independent composition, architecture and properties [2]. Division of plant cells occurs, after the chromosomes have been duplicated and a phragmoplast formed, by the synthesis of a new wall segment splitting the cell into two halves[1].

Cells maintain hydrostatic pressure, or turgor, which produces strain in the walls[1]. Growth is an interplay between this pressure driven stretching of walls and

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 78 – 87, 2005.

biosynthetic processes that modify the walls' protein structure [3]. Enzymes and other agents maintain the mechanical strength and extensibility of cell walls until they cease growing [4]. The two processes occur on different time scales; biosynthesis occurring over hours or days, and elastic stretching over seconds or minutes [3]. The interface between two neighbouring plant cells is typically linear in cross-section (figure 2), suggesting minimal pressure differences between neighbours.

Plant cells are often polar, that is they organise their behaviour along directional axes. Growth and division are coordinated with respect to these axes.

Factors that regulate cell behaviour (e.g. via gene expression), such as hormones, transcription factors and other molecules are transported from cell to cell. These signalling pathways have been shown to play a role in directing many developmental processes including cell shape, growth, movement, and proliferation [5], as well as cell polarity orientation[6, 7].

Signalling and other mechanisms provide key positional information, which coordinates differential cell behaviour. Although lineage plays some role in cell fate specification, it seems positional information is of primary importance in plant development [8, 9].

1.2 Modelling Approaches

Several approaches to modelling cell shape mechanics and multicellular organisation have been developed. Although not particular to plant cells, these models provide a basis on which to elaborate specific techniques for examining plant morphogenesis.
Fleischer and Barr [10] used a spherical geometric representation of cells, concentrating their efforts on cell proliferation and subsequent multi-cellular organisation. They took a rule-based approach to cell behaviour and used diffusion for cell-cell signalling. Although useful, the spherical (or any other primitive-based) model cannot properly capture the range of cell-cell relations that forms the basis for the intercellular signalling that is thought to direct plant growth. In particular, the Fleischer representation limits cells to sphere packing arrangements, which are not typical of plant cells.

Honda et al [11, 12] and Kawasaki and Okuzuno [13] developed 2- and 3-dimensional vertex methods for modelling cell shape. These models describe the cell-boundary as a collection of linear (2D) or polygonal (3D) faces. They considered the effects of cell division in [12], but did not consider cell behaviour in any detail. The linear representation of cell-cell interfaces fits well with observed plant cell shapes, and allows realistic multi-cell arrangements.

We take a hybrid approach, using a similar geometric representation to [12] and [13], with the diffusion signalling used in [10]. We add anisotropic cell behaviour and a system for specifying arbitrary models of genetic regulation.

2 Computational Model

Each cell is defined by a set of dynamic state parameters (including morphogen levels, growth rate, etc.) and a closed boundary. The state of the cell determines its behaviour at any point in time. Cell behaviour is expressed as the transformation of

cell state parameters to proceed to a new state. These transformations are defined in the genetic script, which is the same for each cell.

The boundary of the cell describes its shape, and is decomposed into a set of *walls*. Each wall is the interface between two cells. Morphogens move from one cell to the other via the wall, and both of the adjacent cells have equal influence on its properties. Cell growth is the process of increasing the lengths of the walls to varying degrees.

The state parameters provide feedback between the cell shape, cell-cell signalling, and the regulation of cell behaviour. The nature of the coupling of these feedback processes is arbitrarily defined in the genetic script.
The model is iterated as follows:

1. Execute genetic script for each cell
2. Iterate the morphogen diffusion system over N time-steps
3. Adjust the mechanical properties of the walls
4. Find the equilibrium wall configuration
5. Repeat from step 1

2.1 Cell Shape

We model cell geometry in 2-dimensions – approximating a layer of cells. The cell walls are modelled as *two* linearly elastic elements (springs) bound together at the end points. Each of the adjacent cells influences the properties of only one of these springs. Each spring has stiffness K and natural length L_n determined from the state parameters of the appropriate cell. Simulated forces are computed at each vertex as shown in figure 1A. The magnitude of the turgor force is PL, where P is the cell pressure and L is the wall length.

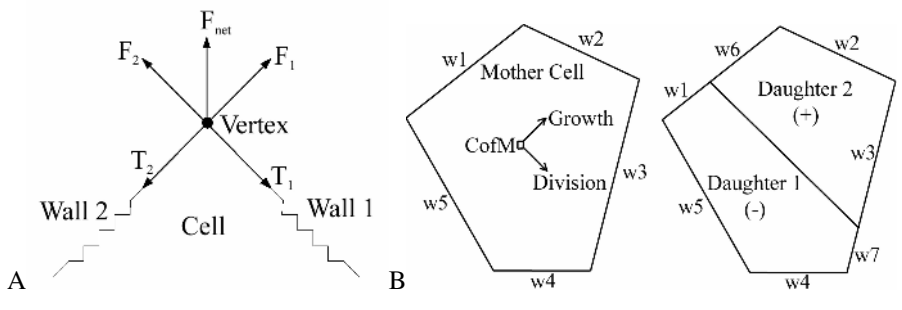


Fig. 1. (A) Forces on a vertex due to turgor (F) from one cell, and elastic tension (T). (B) Cell division consists of inserting a new wall across the centre of mass of the cell. Existing walls *w1* and *w3* are split in two by the division.

Cell growth is achieved by increasing the natural lengths of the springs associated with the growing cell, simulating biosynthesis of wall materials. At each time step the cells' growth rates are used to compute the natural lengths of each of the springs, using:

$$L_n^{t+1} = L_n^t + \lambda(L^t - L_n^t) \tag{1}$$

where λ is a function of growth rate defined on [0,1] (see next section), and superscripts denote time step.

Synthesis of wall constituents continually maintains wall strength [4]. In order to model this we impose the limit on L_n:

$$L_n^{t+1} = Max\left\{L_n^t, L_n^t + \lambda(L^t - L_n^t), (1-\varepsilon_o)L^t\right\} \quad (2)$$

where ε_o is the maximum strain the wall tolerates without reinforcing its structure.

The model assumes that cell growth occurs at a much slower rate than that at which forces propagate through the cell wall matrix. This means that a kind of temporary equilibrium can be assumed at each time step, such that the walls have rearranged themselves so as to minimize the forces on them. We use a Runge-Kutta algorithm [14] to find the equilibrium vertex positions at each time-step.

2.2 Cell Polarity

We maintain a pair of orthogonal unit polarity vectors, $\hat{\mathbf{v}}_1$ and $\hat{\mathbf{v}}_2$, for each cell. These are initialised arbitrarily, and then updated each time step to reflect the rotation of the cell as it grows. Cell growth, division, and the separation of morphogens after division, are each organised about one of the polarity vectors (which one to use is determined by the genetic script).

Cell growth is polarised by defining the function λ in equation 1 as:

$$\lambda = R\left[1 - a\left(\hat{\mathbf{v}} \cdot \hat{\mathbf{w}}\right)^2\right] \quad (3)$$

Where R is the cell growth rate, $\hat{\mathbf{v}}$ is the polarity vector, $\hat{\mathbf{w}}$ is the unit direction of the wall (anti-clockwise with respect to the cell), and a is the degree of anisotropy defined on [0,1]. With $a=0$ we get isotropic growth, and with $a=1$ we get growth mostly in walls aligned closely with the polarity vector $\hat{\mathbf{v}}$.

Cell division is achieved by inserting a new wall positioned to pass through the centre of mass of the cell, and aligned in the direction of the chosen polarity vector. Figure 1B illustrates the process. Two new cells are created on either side of the dividing wall. The morphogen levels of the mother cell are split between the two daughter cells. Each cell receives ½(1+ s_i) (daughter cell 1 in figure 1B) or ½(1- s_i) (daughter cell 2) of the mother cell morphogen, where s_i is the asymmetry factor on [-1,1] defined by the genetic script.

Inheritance of morphogen levels allows the model to capture cell lineage, and asymmetric separation of morphogens makes it possible to consider branching of lineages.

2.3 Genetic Regulation

The genetic script is implemented via an embedded Tcl system. It may perform any sequence of Tcl instructions (e.g. logical conditions, resetting) on the cell state parameters. In addition two procedures are defined. The *divide* procedure instigates

cell division, instructing the spatial model to adjust itself accordingly. The script may also cause the cell to die via a *kill* procedure.

The parameters that define the state of each cell and which are available for transformation by the genetic script are: morphogen production rates, morphogen localisation asymmetry s_i, growth rate R, anisotropy a, growth axis, division axis, and turgor pressure P. Other values are available as *read only* variables: volume V, and morphogen concentrations u_i.

The system is general enough to allow implementation of genetic regulation on many levels, from differential equations to rule-based models.

2.4 Signal Transduction

We consider cell-cell signalling by passive diffusion transport of morphogens produced by the cells. Diffusion allows us to model long-range (high diffusion rate), short-range cell-to-cell (low diffusion rate), and cell-autonomous (zero diffusion rate) signal molecules within the same mathematical framework.

The signalling system is iterated at each cell j for each morphogen i using:

$$dU_i^j(t) = \delta t\left[\gamma_i \sum_{walls}\left(U_i^{neighbour} - U_i^j\right)L + pU_i^j\right] \tag{4}$$

where U_i^j is the morphogen *concentration*, γ_i (units [distance]$^{-1}$[time]$^{-1}$) is the rate of transport per unit length of wall per unit difference in concentration across the wall, L is the length of the wall. The sum is over the walls forming the boundary of cell j and *neighbour* is the cell adjacent to j across each wall. The term pU_i^j is the rate of morphogen production.

3 Results and Discussion

We performed simulated experiments to assess the performance of the model in four key aspects of plant morphogenesis: cell proliferation, coordinated growth, cell lineage, and cell position specification. Parameter settings $P=0.1$, $K=1$ were used throughout.

3.1 Proliferation

Simple cell colonies were generated from initial conditions of a single unit square cell. All cells were grown at the same rate ($R=0.1$) and divided when their volume doubled. Cells inherited polarity from initial vectors: up (axial) and right (lateral). All growth was indeterminate; analysis was limited to the first few hundred time-steps.

Figure 3A shows a colony generated by alternating the cells' division axes and maintaining growth axes perpendicular to it. After each division the growth and division axes were flipped from axial to lateral or vice versa. Cell growth was isotropic. The resulting colony shows regular cell size and shape and its boundary maintains the square shape of the initial cell. The cell walls show the characteristic zigzag wall pattern seen between adjacent cell files in plant roots and shoots (figure 2A).

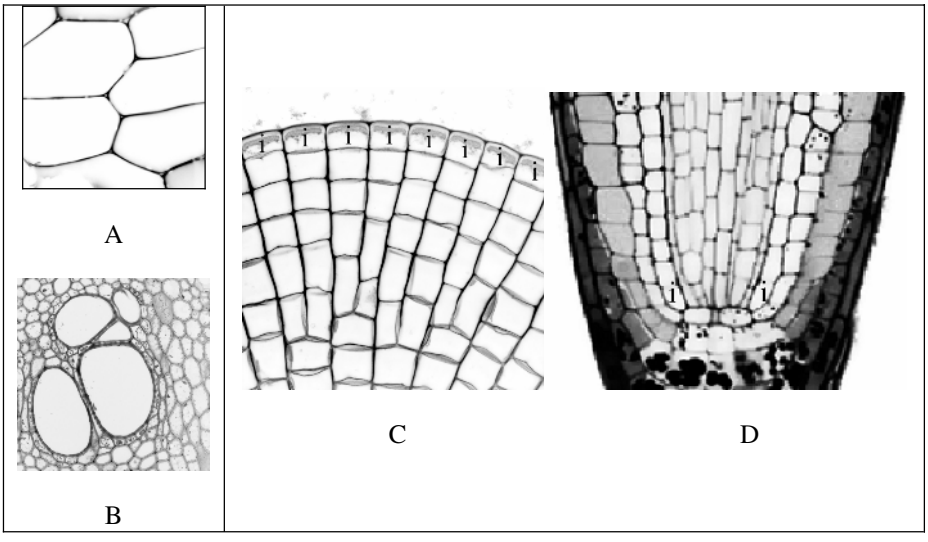


Fig. 2. Examples of cell arrangements in plant tissues. (A) Cell walls show a characteristic zigzag pattern caused by lateral tension forces at T-junctions. (B) Scale range of cell cross-section area, ranging to approximately 1:100. (C, D) Initial or stem cells (i) are maintained at fixed positions in the meristem. (C) Coleochaete scutata (a simple green alga), the stem cells are maintained at the thalus margin. (D) Arabidopsis thaliana root meristem [Reproduced courtesy of Sarah Hodge]), initials remain in fixed positions relative to the root tip after divisions producing other cell types.

Random cell colonies were generated by choosing division axes from a pseudo-random number, with equal probability of axial or lateral division. Again, the cells' growth axes were maintained perpendicular to their division axes. Cell growth was polar, with $a=0.9$.

Figures 3B and 3C show three colonies generated by different pseudo-random seeds. As expected, the cells have a broader range of shapes compared with colonies generated by alternating division. Given equal probability of lateral and axial division, we might expect little average change in colony shape over time. However, overall colony shapes varied widely between random seeds.

The elongated form of figure 3B was caused by constraints on growth imposed by cells on their neighbours. An early sequence of in-line divisions established the long thin shape of the cell colony, which was maintained by the combination of anisotropic growth and coordination of growth by the two-spring model. To illustrate this, consider a line of three cells. If the two end cells are growing along the line and the central cell opposite to the line, growth of the central cell will be retarded. This is because its lateral walls each consist of one growing spring and one non-growing spring. Growth along the line therefore takes place at a much greater rate than lateral growth. In figure 3C similar growth constraints caused an asymmetry in cell-number on the left and right flanks, which was amplified through cell proliferation, causing a dog-leg similar to gravitropism in plant roots.

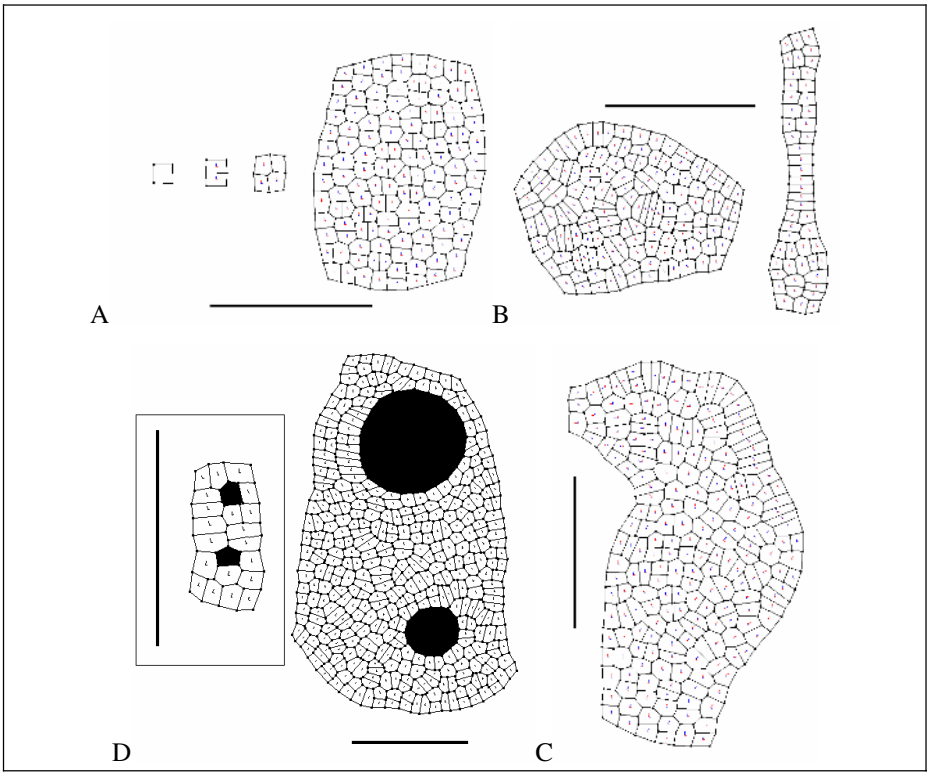


Fig. 3. Indeterminate growth organised by inherited polarity, scale bar shows 10 units. (A) Alternating division axis with growth axis perpendicular: steps 0, 27, 69, and 203 (130 cells). (B,C) Division axis chosen randomly with growth axis perpendicular: clockwise - step 312 (135 cells), step 314 (50 cells), step 390 (210 cells). (D) Growth as in B and C, with two cells (inset) chosen to stop dividing whilst continuing to grow.

Figure 3D shows the effect of manually specifying mitotic inactivity in a few cells. The selected cells (shaded) continued to grow at the same rate but did not divide. The colony showed scale differences in cell size not uncommon in many plants tissues, as illustrated in figure 2B.

3.2 Coordinated Growth

There are several examples of processes in plants (e.g. lateral root development) in which a zone of proliferating cells is established within a mature or slowly growing region. In order to examine this process we triggered proliferation in a single cell and its descendents by injecting a non-diffusing morphogen. The genetic script was configured to trigger growth at $R=0.1$ on presence of this morphogen. Growth was polar ($a=0.9$), and all cells divided on doubling their initial volume.

The effect of the maximum wall strain parameter was examined, keeping all cells turgor pressures equal. Figure 4A shows the results with $\varepsilon_o=0.1$ (approximately the

strain level produced by the cells' turgor). Growth continues indefinitely and surrounding cells are stretched to the point where they divide. With $\varepsilon_o=0.5$ (figure 4B), there is no proliferation and minimal cell growth. These results suggest that zones of varying growth rates could be coordinated via transmission of forces and passive cell wall biosynthesis, without requiring differential turgor.

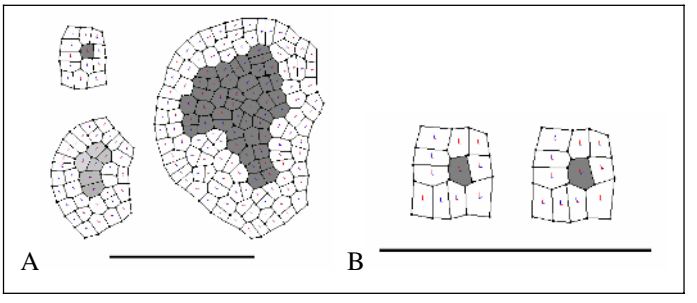


Fig. 4. Cell proliferation zone (shaded) surrounded by non-growing cells (grey tones not significant). Scale bars show 10 units. (A) Cell maximum strain $\varepsilon_o=0.1$; growth is indefinite with surrounding cells stretched: step 0 (13 cells), step 44 (37 cells), step 76 (135 cells). (B) $\varepsilon_o=0.5$, equilibrium reached in right-hand image, cell proliferation constrained.

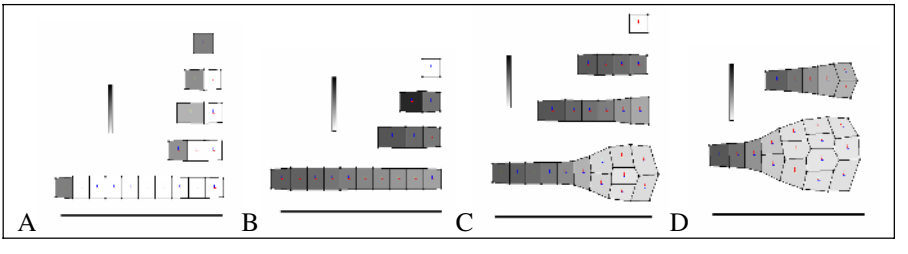


Fig. 5. Morphogens controlling development by inheritance and positional information. In all cases morphogen level shaded from white (zero) to black (one), and scale bar shows 10 units. (A) Inheritance of stem cell character; morphogen inducing 1-D growth and division inherited by only one daughter cell, other cells inactive (steps 0, 156, 242, 311, and 1915). (B) Morphogen gradient; morphogen produced by stem cell, diffuses to other cells where it is degraded (steps 0, 195, 400, 1315). (C,D) Gradient of lateral growth inhibitor; inhibition threshold at 0.5 (C) (steps 0, 461, 832, 1154) and 0.75 (D) (steps 605 and 817).

3.3 Cell Lineage and Positional Information

The relative roles of cell lineage or inheritance, and cell-cell signalling mechanisms and their interactions are important in understanding plant development. We demonstrate both mechanisms in our model independently and in combination.

A stem cell lineage was established using a non-diffusing morphogen with division asymmetry of $s_i=1$ (figure 5A). The morphogen was used to trigger growth and division in 1-dimension. This maintained an active cell at the end of a line of inactive cells, in a similar manner to a plant root- or shoot meristem (figure 2D).

In figure 5B, the stem cell was used to generate a morphogen gradient. The stem cell produced the morphogen shown so as to maintain constant concentration. The morphogen diffused into non-stem cells where it was degraded at a rate proportional to its concentration. The result was an approximately linear gradient along the length of the cell line.

The combination of lineage generated stem-cell and morphogen gradient was used to establish morphogenetic zones along the cell line (figures 5C and 5D). Non-stem cells were scripted to proliferate laterally ($R=0.05$) if the morphogen was below a threshold value. The threshold fixed the distance from the stem-cell up to which lateral growth was inhibited. A higher threshold made the inhibited zone smaller and vice versa. Similar zones of varying growth and division can be identified at characteristic positions across plant meristems (figure 2C,D), and the traffic of plant growth regulators is known to be involved in their delineation.

4 Conclusions

A method has been developed for simulating the particular features of cellular scale plant morphogenesis. The method builds on previous models of cell shape and multicellular organisation with the addition of polar cell behaviour. Although no specific model of genetic regulation has been put forward, a system which can integrate any such model with both a spatio-mechanical and a signalling model has been demonstrated.

The capacity of the system to reproduce key features of plant morphogenesis has been demonstrated using simple rule-based genetic logic. Initial experiments confirm that the model can produce plant-like cell proliferation, coordination of growth zones, and specification of cell behaviour by lineage and position. Thus the system provides a sound basis on which to investigate the complex interactions of all of these elements operating together.

The system has been implemented as interactive software, and as such provides a valuable tool for hypothesis testing in a controlled setting. It also provides a test bed for evaluating models of genetic regulation operating within, as well as controlling cellular development. Work is currently underway on integrating a gene network model into the system, and an evolutionary algorithm for generating networks that produce particular developmental systems.

The results presented here suggest that spatio-mechanical interactions place significant constraints on the shape formation potential of genetic control and patterning. Initial asymmetries in growth were amplified throughout development, and cell growth opposing early growth patterns was constrained. Further work remains to be done in quantifying and analysing these constraints in order to determine the organising potential of mechanical interactions alone, and in combination with genetic regulation in cell colonies.

Acknowledgments

Work at the University of Cambridge was funded jointly by the BBSRC and EPSRC under the E-science program, and by a Sainsbury Fellowship to JH from The Gatsby

Charitable Foundation. TR completed part of this work with funding and support from the Australian Research Council Centre for Complex Systems, and Prof. Kevin Burrage of the Advanced Computational Modelling Centre, University of Queensland. We thank Sarah Hodge for providing assistance with microscopy.

References

1. Alberts, B.: Molecular biology of the cell. 3rd ed., Garland,, New York : (1994) 1294p :
2. Roberts, K.: The plant extracellular matrix: in a new expansive mood. Current Opinion in Cell Biology 6 (1994) 688-694
3. Veytsman, B. A., Cosgrove, D. J.: A Model of Cell Wall Expansion Based on Thermodynamics of Polymer Networks. Biophys. J. 75 (1998) 2240-2250
4. Cosgrove, D. J.: ENZYMES AND OTHER AGENTS THAT ENHANCE CELL WALL EXTENSIBILITY. Annu. Rev. Plant Physiol. Plant Mol. Biol. 50 (1999) 391-417
5. Cummings, F. W.: A Model of Pattern Formation Based on Signalling Pathways. J. Theor. Biol. 207 (2000)
6. Weijers, D., Jurgens, G.: Auxin and embryo axis formation: the ends in sight. Current Opinion in Plant Biology 8 (2005) 32-37
7. Goldstein, B.: When Cells Tell Their Neighbours Which Direction to Divide. Developmental Dynamics 218 (2000) 23-29
8. Dolan, L., Roberts, K.: Plant development: pulled up by the roots. Current Opinion in Genetics and Development 5 (1995) 432-438
9. Fleming, A. J.: The mechanism of leaf morphogenesis. Planta 216 (2002) 17-22
10. Fleischer, K., Barr, A. H.: A Simulation Testbed for the Study of Multicellular Development: The Multiple Mechanisms of Morphogenesis. In Langton, C., ed.: Artificial Life III Redwood, CA, Addison-Wesley (1994) 389-416
11. Honda, H., Tanemura, M., Nagai, T.: A three-dimensional vertex dynamics cell model of space-filling polyhedra simulating cell behaviour in a cell aggregate. Journal of Theoretical Biology 226 (2004) 439-453
12. Honda, H., Yamanaka, H., Dan-Sohkawa, M.: A computer simulation of geometrical configurations during cell division. Journal of Theoretical Biology 106 (1984) 423-435
13. Kawasaki, K., Okuzuno, T.: Computer Simulation of Cellular Pattern Growth in Two and Three Dimensions. Phase Transitions 28 (1990) 177-211
14. Pozrikidis, C.: Numerical Computation in Science and Engineering. Oxford University Press, New York, Oxford (1998) 475-479

A Developmental Model for Generative Media

Jon McCormack

Centre for Electronic Media Art,
School of Computer Science and Software Engineering,
Monash University,
Clayton, Victoria 3800, Australia
jonmc@csse.monash.edu.au

Abstract. Developmental models simulate the spatio-temporal development of a complex system. The system described in this paper combines the advantages of a number of previously disparate models, such as timed L-systems and cellular programming, into a single system with extensive modeling flexibility. The new system includes the ability to specify dynamic hierarchies as part of the specification, and a decoupling of cell development from interpretation. Examples in application areas of computer animation and music synthesis are provided.

1 Introduction

We are interested in generalized models that simulate the continuous development of some complex system in space-time. This paper describes a new developmental system for the dynamic simulation of organic forms and processes. By decoupling the generative process from the generated output, dynamic models can be created in a variety of different application domains, including biological and botanical simulation, music composition, interactive animation, and computer graphics.

The developmental system described in this paper is strongly influenced by related work in developmental modeling using L-systems, in particular *parametric*, *timed* and *differential* L-systems. The original formulation of L-systems by Lindenmayer in 1968 was a conceptually elegant, discrete, symbolic model of development in cellular biology [1]. In 1990, Lindenmayer and Prusinkiewicz published *The Algorithmic Beauty of Plants*, with an emphasis on three-dimensional, visually realistic models of herbaceous plants [2]. This introduced a number of variations and extensions to L-systems in order to overcome the discrete, symbolic nature of basic constructs such as D0L-systems[1]. Subsequent developments incorporated other continuous developmental control, such as the use of differential equations to model growth and signaling in modules [3] and the effects of environmental constraints [4]. More recent work uses Chomsky grammars in combination with interactive curve editing software to obtain greater visual modeling flexibility and control [5].

Timed L-systems were proposed by Prusinkiewicz and Lindenmayer [2] as an extension for modeling continuous development with D0L-systems. Their description

[1] D0L-systems are *deterministic* and *context free.*

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 88 – 97, 2005.

was limited to D0L-systems without parameters and they restricted their modeling examples to the development of the simple cellular structure of *Anabaena catenula.* The cellular developmental model described in the following section follows naturally from *timed, parametric*, *stochastic* L-system models developed by the author. Further details on these specific extensions can be found in [6, 7].

In parallel with these developments, a number of cellular models of development have been proposed. The cellular model of Fleischer and Barr [8] modeled developing cells that exchanged chemicals by diffusion with simple 'cell programs'. This research was applied to areas such as three-dimensional texture synthesis [9]. More recent work combines related cellular development models with evolution to create structures that grow into target shapes [10].

One area that these systems have difficulty in addressing is in the design of a hierarchy – a mechanism often observed in natural systems. Additionally, they are typically designed for some specific simulation or application and are not necessarily applicable to generalized development. The system described in this paper addresses these issues. The following sections describe the model in detail.

2 The Cellular Developmental Model

2.1 Cell Definitions

In developing this system, we will consider a basic automaton, which is referred to as a *cell*. The name is used as a metaphorical interpretation of cells as found in biological life. The model is 'biologically inspired', but is not designed to reflect a literal interpretation of cellular development. This cellular abstraction is capable (as a simulation) of functions a biological cell does not have, and reciprocally, the biological cell is capable of many functions not possible with the model described here. Cells exist in an abstract entity called the *world*. The world is responsible for the *creation*, *removal*, and *interpretation* of cells that exist within it. Details of these terms and the world itself will be discussed shortly.

A cell is composed of four principle components (refer Fig. 1):

- A *label*, $s \in V_T$, where V_T is an alphabet that is specific to the cell type (approximately corresponding to the single alphabet of an L-system). The type of a cell is distinguished by its ability to develop or be interpreted;
- A *state*, $\Sigma_T \in \left(\Re^* \times \Im^*\right)$ — a set of variables that reflect measurable properties (both internal and external) that the cell possesses. The state will change dynamically subject to the mechanisms of the cell;

  A set of predicate *rules* or *productions*, $P_T \subset \left(V \times \Sigma^*\right) \times C(\Sigma) \times \left(V \times E(\Sigma)^*\right)^*$, where $C(\Sigma)$ and $E(\Sigma)$ are respectively the set of logical and arithmetic/functional expressions using parameters from Σ. Rules specify developmental changes to the cell, and may consider the cell state as well as the state of *neighbouring* cells (the concept of a neighbour is defined shortly);

- An *interpretation*, $I \subset \left(V_I \times E(\Sigma_T)\right)^*$, which is a set of instructions as to how the cell is to be realised in the world (V_I is the alphabet of a particular set of interpretative symbols). The interpretation can make use of the cell's state.

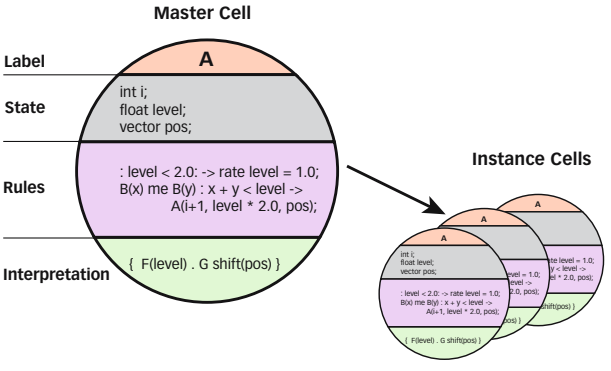


Fig. 1. Master and instance cells, and their principle components

Cellular instantiation follows the class/object model used in object-oriented programming [11][2]. Cells are instantiated into *pools* (spatial data structures), and there may be many *instance* cells with the same label, but each cell carries its own state, which develops independently. Conceptually, each cell also carries its own copy of the rules and interpretation defined for a cell of that label, although in most situations these are references to the rules and interpretation contained in the *master cell*. Thus, normally no distinction needs to be made between master cells and instance cells.

A special type of cell is called a *system*. A system has a label and contains state information, but does not have any rules or interpretation. Unlike a normal cell, a system cell may contain other cells, including other system cells. Thus, the system cell is capable of forming a hierarchical structure (Fig. 2). Systems contain an *initial state* (or *axiom*) that consists of a sequence of instance cells with particular state initialization information. They also maintain a pool wherein cells may be created, replaced, and deleted. A *root system* contains all other cells and systems and is created automatically by the world upon initialization. The root system's age will automatically reflect the developmental time of the entire system.

System cells may contain other cells (which may be systems too), but they cannot contain instances of themselves, nor can sub-systems contain instances of parent cells. This ensures the cellular hierarchy maintains a tree structure (rather than a cyclic graph, which would permit illegal circular definitions). A hierarchical structure is a good way of describing many natural patterns and forms [13].

Cells within a system develop asynchronously, however cells *may* synchronize development based on examination of each other's state. In addition, cells have access

[2] Alan Kay, inventor of the *Smalltalk* programming language, used biological metaphors in its design, likening the concept of objects to 'cells' with walls, with the class providing a well-defined boundary between co-operating units [12].

to the state of any parent cells, including the state of the root system. Specific components of the cell will now be described in more detail.

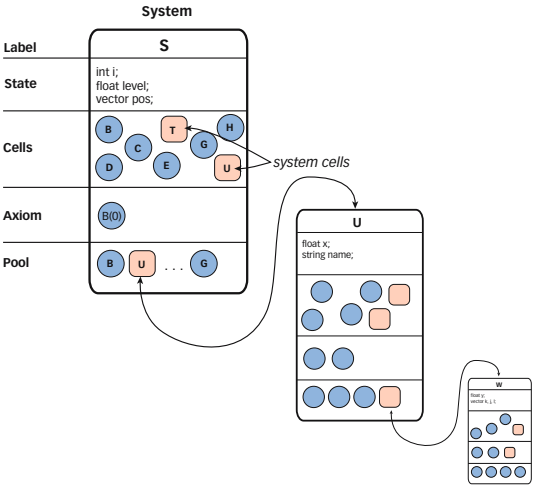


Fig. 2. A system cell may contain other cells, which in turn may be system cells. Thus the cellular hierarchy is formed.

Cell State

Cell state captures the measurable components of a cell. The state is a vector composed of both user- and system-defined quantities. The user may define the cell state as required by the cell's particular type. In addition, all cells maintain a number of internal states that are defined and managed by the cell itself. Internal states are 'read-only' — available as symbols for use in productions, but they cannot be modified, hence they provide an introspection of various fixed components of the cell. The internal state includes the cell *age*, a continuous scalar that is automatically updated to reflect the age of the cell during its lifetime. An internal *status* contains four discrete states affecting overall cell behaviour: (i) *dormant* where no state changes are effected (the usual state of master cells); (ii) *birth* where state is initialized and the cell appears in its parent system's pool; (iii) *alive* where state development proceeds continuously (e.g. states such as the cell's age are continuously updated); (iv) *dead* where the cell will be removed from the current pool it resides in.

Cell Rules

Cellular rules, denoted r_i are ordered sets of *predicate-action* sequences of the form:

$$r_i : \underbrace{\{context\} : predicate}_{\text{predicate component}} : \underbrace{(state\ calculations \mid cell\ actions)}_{\text{action component}} \tag{1}$$

Rules are numbered implicitly in the order of their declaration. While a cell is in the *alive* state, its set of rules is evaluated in ascending order. For a rule to be considered, first, the context requirements must be satisfied. Following this, if the predicate

component evaluates to TRUE (non-zero), the action component is executed. The action component may consist of calculations that change the cell state, or actions that the cell should perform. The following sections describe each component in more detail.

Context

A context statement involves the position in the pool of the current cell in relation to other cells. A special reserved word, me, is used to represent the current cell. This allows cells with the same label to be used in context specifications. Context statements also specify the public state variables of cells involved in the context specification, and these identifiers may then be used in state calculations involving the current cell, i.e. a cell may update its state based on the state of its neighbours. Access to the state of neighbouring cells is read-only. Changing another cell's state directly is not permitted.

Here is an example context sensitive rule, where a cell maintains a component of its state to be the average of its neighbours[3], provided it exceeds some minimum threshold:

$$A(y)\ me\ B(z) : y > k_{\min}\ \&\&\ z > k_{\min} : x = \frac{y+z}{2} \tag{2}$$

which assumes the cell that owns this rule has a state variable, x. The rule first checks if the context is satisfied — that cells with labels 'A' and 'B' are at the 'left' and 'right' of the current cell. If that relationship is TRUE, the state parameters are then checked to see if they exceed some minimum constant value ($k_{\min}$), if so the current cell's state variable x is updated to be the average of the values of it's neighbours.

Context relations have a more flexible meaning than with context sensitive L-systems where the derivation string is a one-dimensional array, and so context matches are decided on by matching symbols to the left and right of the current symbol in the derivation string (a one-dimensional context). The pool in which the cells exist is designed in an abstract way, where the interpretation of neighbour relationships is flexible. This is achieved using polymorphic functions to match context based on pool type. It is important to match context dimension to pool dimension (e.g., a two-dimensional context relation makes no sense to a one-dimensional array).

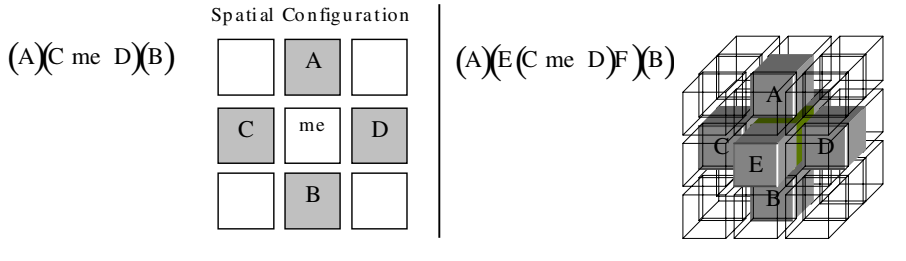


Fig. 3. Higher dimensional context relations and their specification

[3] This example uses a one-dimensional context relation; higher dimensional relations are defined in this section.

In the cellular developmental system, context may include other *spatial* relationships where context relations are satisfied when the spatial position of the cell is less than some Euclidian distance, or topological relationships such as the Von Neumann neighbourhood (Fig. 3) used in cellular automata simulations [14]. The use of parenthesis demarks dimensions when specifying context.

As the system described here uses polymorphic objects to represent the pool (e.g. linear set, multi-dimensional array, spatial structure), the interpretation of context is determined by the way the specific pool interprets the context statements. This permits flexibility in the types of simulations the system can perform. For example, such context relationships can be used in music generation where context relations work in two dimensions: pitch and time (hence context matching can be with chords, rather than notes.

State Calculations and the Differential Operator

State calculations are mathematical expressions that affect a cell's state. They are expressed in a similar manner to expressions in the C programming language. A rich set of functions is provided, including basic mathematical functions, trigonometric functions and a variety of stochastic and noise functions. A unary differential operator, `rate`, performs a specific form of ordinary differential equation solving with initial values. The operand for rate is a local cell state variable. The variable is integrated based on its initial value at birth according to the mathematical formula specified in the state calculation. The differential operator is useful for simulating processes such as chemical diffusion between neighbouring cells.

Actions

Actions correspond to the re-writing process in L-systems, being similar to the successor word of L-systems. However, the concept of rewriting can be misleading, due to the way cell actions are handled and cells are placed in the pool. In the case of L-systems, symbols are always *replaced* by rewriting productions. Whereas, for the cellular developmental model, in addition to replacement, cells may continue to exist in the pool while their actions cause new cells to be added (i.e. the cell action does not necessarily replace (rewrite) the cell instigating that action). Possible actions are outlined in Table 1.

Table 1. Rule actions

Type	Description	Example action syntax
None	No action (the current cell remains)	$\rightarrow me$
Replace	The current cell is replaced	$\rightarrow A(x,y)$
Add	New cells are created and the current cell remains in the pool.	$\rightarrow A(x,y)\ me\ B(x+y)$
Delete	The current cell is deleted	$\rightarrow \varnothing$ (empty string)

2.2 Cell Interpretation

Thus far, cells have been considered in abstraction, without any method of realising them in the world. The interpretation component specifies how the cell will be inter-

preted as the system develops. In the case of L-systems, developmental words are interpreted by a turtle, which creates geometry as it interprets the list of symbols. The case here is similar, with the exception that each cell may contain multiple instructions to the turtle, permitting individual cells to create more complex geometry without polluting the cellular developmental model with cells used only as part of some complex geometry building sequence.

The interpretation system is extremely flexible, in the sense that interpretation contains a list of special cells representing instructions. These cells are of a different type than normal developing cells (they do not develop), but may have associated parameters. These parameters may be set with expressions involving the cell's state. If a cell's state is changing over the lifetime of that cell (the *age* component for example), then the interpretation permits the ability to animate the parameters of interpretive instructions as the cell state changes. This is a more flexible and general form of the development functions associated with timed L-systems [7]. There is no direct dependence between cell development and interpretation, so a number of different interpretative sets can be used on the same developmental system. For example, a musical interpretative set issues musical instructions rather than geometric building ones.

This flexibility allows users of the system to realise their developmental system in a variety of ways, without the need to completely re-specify the grammar. The use of multiple instruction sequences in a single cell is a different solution to a similar problem encountered by Prusinkiewicz and colleagues in the development of their interactive system to model plants [5]. Here they combined a C-like programming language and Chomsky grammars to enable sequential rewriting of strings, rather than the parallel development specified by L-systems.

3 Examples

Complex structures are often modeled by decomposing them into a hierarchy. In this section, an example of this method is shown for developing a model of a multi- segmented, articulated animal with a walking gait. By using a hierarchical descrip-

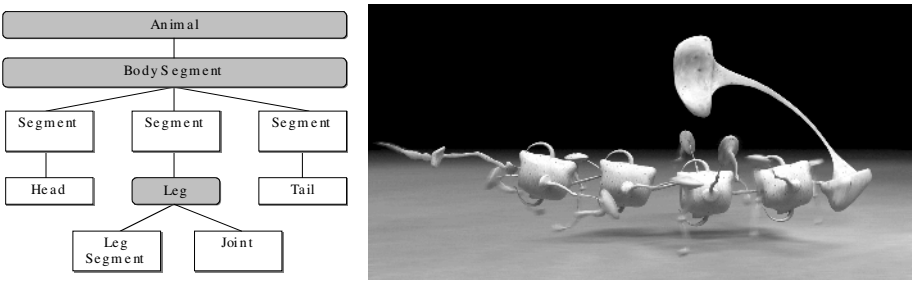


Fig. 4. Hierarchical specification of system cells (grey boxes) and module cells (left); the articulated creature in motion (right)

tion, it is possible to specify structure in an intuitive way. The hierarchical structure of the animal is shown in Fig. 4.

This structure is specified using the cellular programming language, containing cell definitions in a human readable form. The system simulates forward kinematics, with locomotive drivers at the joints to create legged gaits. The drivers function like a state machine controlling gait movement in each leg. The key advantage of the developmental system is in the flexibility of specification, permitting morphological changes to be made easily. For example, the number of body segments is controlled by a single parameter. Geometry is constructed using generalized cylinders; the shape and configuration controlled by lower-level cells driving geometry construction. A resultant still from the animated output of this system is also shown in Fig. 4.

4.2 Music Generation

Interpretation of cell states is not limited to geometric constructs. Using an object-oriented approach allows the interpretation of cell states by methods other than a turtle interpretation. To provide different interpretations a collection of global modules are imported into the namespace of a particular system (the root system by default). These modules are directly interpreted as generative commands.

The musical commands use the concept of a state-based *player*, which is responsible for converting commands into actual music. The player maintains a state that includes the current pitch (note) and volume. The player converts incoming commands into midi[4] messages, enabling any midi compatible device to play music generated by the system.

4 Summary

The developmental system described here unifies a number of previous L-system and cellular models in the application domain of temporal developmental systems. Based on both discrete cellular changes and continuous state development, the model successfully integrates these two modes of development, and permits complex temporal sequences not achievable using previous techniques.

The hierarchical nature of the cellular developmental system allows management of complexity from the point of view of the user specifying a system model. Hierarchical ordering increases the control over structure at variety of levels, hence reducing the 'brittleness'[5] of a flat grammar specification. This permits a more intuitive control over creation of modeled systems. As show in Fig. 5, the complex nature of cell development stands in contrast to the more simplified discrete D0L-system model.

---

[4] MIDI is a low-level serial communication protocol for musical instruments and musical controllers — see [15].

[5] That is, more robust to configuration changes — a change at one level in a hierarchy can have fewer side effects than for the equivalent description in a non-hierarchical system.

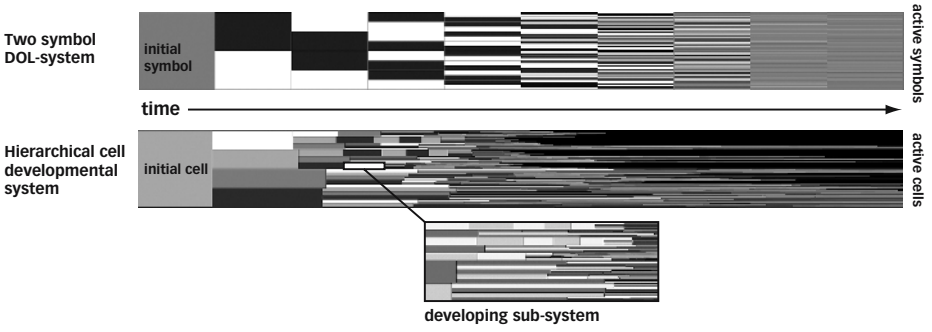


Fig. 5. Time-state diagram contrasting the developmental differences between discrete L-systems and the cellular developmental system described in this paper. The diagram shows the temporal development (from left to right) of each system. Shaded rectangles represent the symbols present (vertical axis) at any given time (horizontal axis). Both examples start from a single symbol or cell (the axiom). In the case of the DOL-System (top) each iteration is clearly synchronized and regular as the rewriting process proceeds in discrete time steps. In the case of the developmental cellular system, the sequence quickly becomes irregular and 'fractal' due to the individual developmental nature of cells. A single cell at the top level is shown expanded as a segment of a developing sub-system, illustrating the complexity that is contained by the hierarchy.

References

1. Lindenmayer, A.: Mathematical Models for Cellular Interactions in Development, Parts I and II. Journal of Theoretical Biology 18 (1968) 280-315
2. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. The Virtual Laboratory. Springer-Verlag New York (1990)
3. Prusinkiewicz, P., Hammel, M., Mjolsness, E.: Animation of Plant Development. Proceedings of SIGGRAPH 93 (Anaheim, California, August 1-6, 1993) In Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1993) 351-360
4. Prusinkiewicz, P., James, M., Mech, R.: Synthetic Topiary. Proceedings of SIGGRAPH 94 (Orlando, Florida, July 24-29, 1994) In Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, (1994) 351-358
5. Prusinkiewicz, P., Mündermann, L., Karwowski, R., Lane, B.: The Use of Positional Information in the Modeling of Plants. Proceedings of SIGGRAPH 2001 (Los Angeles, California, August 12-17) In Computer Graphics Proceedings Annual Conference Series, ACM SIGGRAPH, (2001) 289-300
6. McCormack, J.: The Application of L-Systems and Developmental Models to Computer Art, Animation, and Music Synthesis. Ph.D. thesis, School of Computer Science and Software Engineering, Monash University, Clayton (2003)
7. McCormack, J.: Generative Modelling with Timed L-Systems. In Gero, J.S. (ed.) Design Computing and Cognition '04, Kluwer Academic Publishers, Dordrecht (2004) 157-175
8. Fleischer, K.W., Barr, A.H.: A Simulation Testbed for the Study of Multicellular Development: The Multiple Mechanisms of Morphogenesis. In Langton, C.G. (ed.) Artificial Life III, Addison-Wesley, Reading, Massachusetts (1994) 389-416
9. Fleischer, K.W., Laidlaw, D.H., Currin, B.L., Barr, A.H.: Cellular Texture Generation. Proceedings of SIGGRAPH 95 (Los Angeles, California, August 6-11, 1995) In Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, (1995) 239-248

10. Kumar, S., Bentley, P.J.: Mechanisms of Oriented Cell Division in Computational Development. First Australian Conference on Artificial Life (ACAL 2003). Canbera, Australia (2003)
11. Wirfs-Brock, R., Wilkerson, B., Wiener, L.: Designing Object-Oriented Software. Prentice Hall Englewood Cliffs, N.J. (1990)
12. Kay, A.C.: The Early History of Smalltalk. The second ACM SIGPLAN conference on History of programming languages. Cambridge, Massachusetts, ACM Press (1993) 69-95
13. Simon, H.A.: The Sciences of the Artificial. 3rd Edition. MIT Press Cambridge, Mass. (1996)
14. Berlekamp, E.R., Conway, J.H., Guy, R.K.: Winning Ways for Your Mathematical Plays. Vol. 2. Academic Press New York (1982)
15. Selfridge-Field, E. (ed). Beyond MIDI: The Handbook of Musical Codes. MIT Press Cambridge, Massachusetts (1997)

Evolutionary Simulations of Maternal Effects in Artificial Developmental Systems

Artur Matos, Reiji Suzuki, and Takaya Arita

Graduate School of Information Science, Nagoya University,
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
amatos@create.human.nagoya-u.ac.jp, {reiji,arita}@nagoya-u.jp

Abstract. Maternal influence on offspring goes beyond strict nuclear (DNA) inheritance: inherited maternal mRNA, mitochondria, caring and nurturing are all additional sources that affect offspring development, and they can be also shaped by evolution. These additional factors are called maternal effects, and their important role in evolution is well established experimentally. This paper presents two models for maternal effects, based on a genetic algorithm and simulated development of neural networks. We extended a model by Eggenberger by adding two mechanisms for maternal effects: the first mechanism attempts to replicate maternal cytoplasmic control, while the second mechanism replicates interactions between the fetus and the uterine environment. For examining the role of maternal effects in artificial evolution, we evolved networks for the odd-3-parity problem, using increasing rates of maternal influence. Experiments have shown that maternal effects increase adaptiveness in the latter model.

1 Introduction

Traditionally, when considering traits' variation in organisms, sources for variation are divided into genetic contributions and effects due to the environment. Recently, another important source for variation has been increasingly considered, that occurs when the environment for the organism is provided by another (usually con-specific) phenotype. Indirect genetic effects [7] occur when this environmental influence is genetically based, that is, the genes of an individual affect another individual indirectly through the provided environment. Among these effects, maternal effects, that is, effects that occur between mother and offspring are the most extensively studied. Maternal effects are ubiquitous in metazoans and also found extensively in plants. Besides supplying half of the DNA to their offspring, mothers additionally contribute essential factors for their early development, nutrition, rearing, and cultural conditioning [6]. For instance, early developmental stages in all metazoans are under exclusive control of maternal gene products deposited in the egg during oogenesis (egg formation). Even after the transition to zygotic gene regulation, products resultant from early maternal cytoplasmic control still take important roles in development. Additionally, in mammals, interaction between the placenta and the fetus can be an important influence. All of these factors can be additional sources for affecting the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 98–107, 2005.

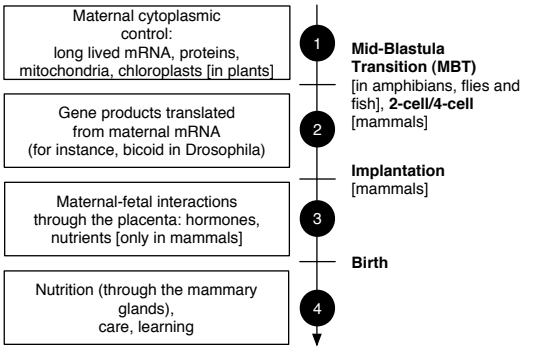


Fig. 1. Summary of the maternal contributions to offspring development

offspring's phenotype in addition to strict nuclear inheritance. A summary of the major maternal effects are shown in figure 1.

Maternal effects can influence evolution significantly in two ways, both producing unusual outcomes. First, contrary to what is expected from standard Mendelian genetics, traits in the mother and in the offspring may be negatively correlated [5]. The end result from this is that selection for a specific trait may in fact produce a temporally reversed response, that is, selection for a larger trait in mothers would produce offspring with smaller traits, and selection for smaller traits would produce larger traits. This was observed experimentally by Falconer [3], who performed artificial selection experiments for larger litter size in mice: In his experiments, selection for larger litter size resulted in mothers having larger litters, but that developed into smaller adults; in contrast, mice from smaller litters would grow into bigger adults, when compared to the ones developed from the larger litters. Falconer inferred that this should be due to increased competition for milk in the larger litters, that would create smaller adults due to less available milk from their mothers.

Second, maternal effects can create time lags in selection response, generating a kind of "evolutionary momentum", where a trait may continue evolving even if selection ceases. Kirkpatrick and Lande [5] created a quantitative genetics model, taking into account maternal inheritance, where this effect is observed. In their model, evolutionary momentum occurs whenever traits between mother and offspring are related, with either a trait present in the mother directly affecting the same trait in offspring, or indirectly through other traits. The direction of evolutionary momentum, or how the affected trait evolved after selection ceases, depends on whether the traits are negatively or positively correlated.

In this article, we introduce two models for maternal effects, both focusing on how these effects occur at the molecular level, i.e. due to exchange of gene products between mother and offspring, and resultant affected gene regulation (Figure 1, boxes (1) and (3)). The first model attempts to replicate maternal cytoplasmic influence in the early stages of metazoan development, while the second models the exchange of chemicals between the mother and the fetus

in mammals. For this, we adopted a developmental model by Eggenberger [2], that uses simulated gene regulation and cell communication for generating neural networks. Coupled with a genetic algorithm, we then evolved networks for solving the odd-3-parity problem, with increasing maternal influence in both models.

2 The Models

2.1 Overview

A conceptual overview of both models, compared with the one based on the standard evolutionary perspective, is shown in figure 2. In these models, both mother and offspring undergo development, that maps their genotype to their final phenotype. There are two sources accounting for variation in the final offspring phenotype, the first being the standard maternal genetic contribution, and the second being an additional mechanism, depending on the model.

The first model attempts to replicate maternal cytoplasmic control as it occurs in the early stages of development in all metazoans. During metazoan oogenesis, the mother places mRNA and proteins in the egg, that directs early development until transcription from the zygote (the fertilized egg) starts. The stage where this occurs depends on the species, being the Mid-Blastula Transition (MBT) stage in amphibians, flies and fish, and the 2-cell or 4-cell stage in mammals. Depending on the species, in these early stages, there can be a significant interaction between the maternal gene products and the zygote, but there are at least some species where maternal control is exclusive. In these species, the eggs are able to develop normally in the early stages, even if the sperm and zygotic nucleus are artificially removed. In a similar way, we decided to model this early maternal control by using the mother's genotype as the exclusive source for the offspring's early development. After *mt* time steps are reached, the real offspring genotype is used for resuming development until *dt* time steps, where it is considered the final phenotype. Using the maternal genome directly, as it occurs in this model, is a significant abstraction from the way real cytoplasmic control works, because maternal mRNA, responsible for early development, is a product from maternal genes, and not the maternal genes themselves. We believe

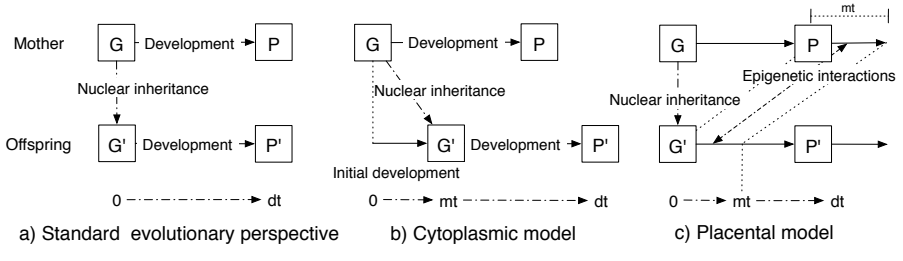


Fig. 2. Conceptual overview. (a) Standard evolutionary perspective. (b) Cytoplasmic model. (c) Placental model.

that this abstraction is still able to preserve the two essential points for this kind of maternal influence, namely: 1) Early development is controlled by maternal genes; 2) It is mainly unidirectional, occurring from the mother to the offspring.

The second model attempts to model interactions between the mother, the growing fetus and the placenta, as it occurs in mammals. In mammals, placenta development occurs after the embryo is formed, and is regulated by both the mother and the embryo. Placental influence works both ways, not only affecting the embryo, but also affecting the mother herself — for instance, by adapting her morphology to nurture adequately after giving birth. Influence in fetus development has been extensively studied in embryo transfer experiments: for instance, Cowley [1] performed embryo transfers between two different mice strains, one significantly larger than the other. In his experiments, mice transfered into the larger strain were always able to grow larger, regardless of their own genotype. An example of influence in the mother are the hormones oestrogen and prolactin, produced by the placenta, and that are responsible for preparing the breasts for milk production in humans. For the sake of simplicity, in our model we decided not to add the placenta as a mediator, and instead to allow the mother and offspring to influence each other directly. In our model, therefore, development occurs concurrently for the mother and offspring; during offspring development, the mother's development is resumed, and epigenetic interactions occur between them for *mt* time steps. This still allows for changes both in the mother and offspring as described before, without having to model an additional entity.

2.2 Development and Evolution

For implementing the models described before, we used a simulated developmental model for neural networks, based on gene regulation and cell communication. Our developmental model is based on a previous one by Eggenberger [2]. We used boolean neural networks with thresholds as either 0 or 1, and the connections being either -1 or 1. Neurons are activated if the sum of the values on their incoming connections is above their threshold. Development proceeds in a rectangular grid, each slot possibly having a neuron, depending on the experiment settings. Each neuron contains a copy of the same genome. Chemicals are generated by gene activity inside each neuron, and diffuse through the grid. Although all neurons contain the same genome, different parts of the genome may be activated on each neuron, due to interactions through chemical diffusion.

All the simulated substances contain a real valued *geom* parameter, with possible values ranging between 0 and 1. This *geom* parameter is used for describing the geometric properties of the substance (for instance, as an abstraction for protein structure), and for attributing a binding between two substances. This is used, for instance, for computing the binding of substances to regulatory regions, and also between Cell Adhesion Molecules (CAMS) as it will be explained later. The affinity between two substances $affinity(geom_1, geom_2)$ is computed by $e^{-|geom_2 - geom_1|}$, where $geom_1$, $geom_2$ are the *geom* values for the two substances. If this affinity is 1 for any two given substances, then they match evenly, while the minimum value, $\frac{1}{e}$, represents no match at all.

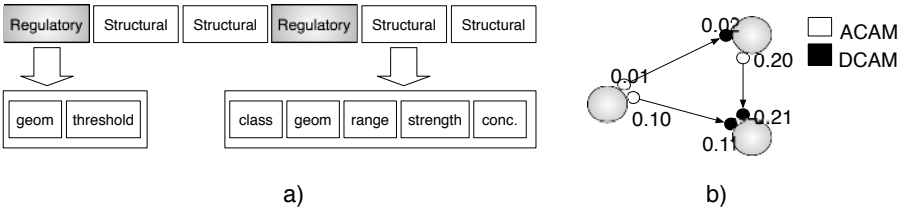


Fig. 3. a) Sample genome structure. In the structural regions, **class** indicates the kind of substance that is produced if the gene is activated, while **geom** represents the geometry of the generated substance. **Range** and **strength** are only expressed in the CAMs: **range** indicates the search range for the CAMs, while **strength** indicates the strength of the synapse (-1 if less than 0.5, 1 if greater than 0.5). **Concentration** indicates the amount produced on each time step, if the gene is activated. As for the regulatory regions, **geom** is used for template matching with the molecules, while **threshold** represents the minimum required affinity for activating the structural gene. b) Neurons connecting to each other. Due to gene regulation, each neuron expresses CAMs in their surfaces, with different **geom** values. Connections are established between (ACAM, DCAM) pairs that have the strongest $\mathit{affinity}(\mathit{geom}_1, \mathit{geom}_2)$.

The genome is real-valued, and organized in an operon-like structure with structural and regulatory regions. Structural regions are responsible, if activated, for generating chemical substances; regulatory regions are used for activating the associated structural regions, depending on the substance's concentration in the cell. A sample structure is shown in figure 3 a).

On each developmental time step, substances inside the neuron compete for binding in the regulatory regions of the genome. The one with the highest affinity to the corresponding regulatory part successfully binds to it. If their concentration times the affinity is above the threshold, then the corresponding structural genes are activated. One regulatory region controls several structural regions, with the number of structural regions per regulatory parts fixed, and defined as a parameter for the experiments. Structural genes can produce three kinds of substances: transcription factors remain inside the neuron, signaling molecules diffuse out of the neuron, and CAMs are used for connecting neurons with synapses. Diffusion is simulated by using a discrete version of Fick's law, with the same diffusion and evaporation coefficients for all the substances. The diffusion equation is:

$$c(x,y,t+1) = (1-4D-E)\cdot c(x,y,t) + D\cdot(c(x-1,y,t) + c(x+1,y,t) + c(x,y-1,t) + c(x,y+1,t)), \quad (1)$$

with $c(x,y,t)$ the concentration of a substance at position (x,y) in the grid at time t, D the diffusion rate, and E the evaporation rate. Concentrations at the boundaries are assumed to be 0. CAMs are expressed at the surface of each neuron, and used for connecting them. Besides the *geom* parameter, they also contain a *strength* and a *range*. They are further divided into two different

kinds: ACAMS (axom) and DCAMS (dendrite). If two cells express CAMs with enough affinity between them, then a connection is established. On each time step, neurons with expressed ACAMS will search on their neighborhood for neurons with suitable DCAMs. If the affinity is high enough, a connection will be established between the pair with the highest affinity, from the ACAM to the DCAM. The search range for each neuron is encoded in the *range* parameter, as a percentage of the whole grid size. *strength* specifies the strength of the connection. An example is shown in figure 3 b).

For simulating evolution, we use a genetic algorithm coupled with the developmental model defined before. No crossover operation was applied, only reproduction and mutation were used. Therefore, the individuals between subsequent generations are always connected by a reproduction operation, and alternate roles between mother and offspring in each generation: that is, an individual in generation m connected to another individual in generation $m + 1$ takes the maternal role for that individual in generation $m + 1$.

Figure 4 shows how both models are implemented. For simulating early cytoplasmic control, development occurs in two discrete stages: in the first stage, the maternal genome is used exclusively in all the cells of the grid until mt time steps are reached. Afterward, the offspring's genome replaces the previous genome in all the cells and guides the remaining development. For the placental model, development for the mother is resumed, and occurs concurrently with the offspring, during mt time steps. During this stage, chemicals are exchanged between the mother and offspring, on each corresponding cell in both grids, using a mD exchange rate. After this stage, development occurs for the offspring as usual, without any further maternal influence.

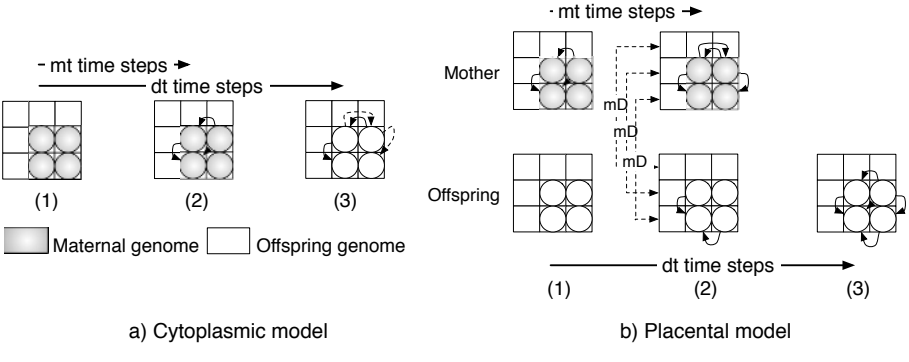


Fig. 4. Diagram for both models. In both models, **(1)** represents the initial stage, **(2)** the stage where until maternal influence occurs, and **(3)** the final stage. Please note that in the placental model, all maternal cells exchange chemicals with their corresponding offspring cell, although this is only shown for the the leftmost column.

3 Experiments and Results

Using this model, we evolved networks for the 3-odd-parity problem. The solution is defined as a neural network with at least 3 inputs, that outputs *true* whenever the number of *true* inputs is odd. All the grids were initialized with a neuron configuration sufficient for this problem: 3 input neurons, 5 hidden neurons (3 with threshold 0, 2 with threshold 1), and 1 output neuron.

A fitness function based on the number of wrong outputs did not produce a good performance so we used a fitness function used by Gruau in [4]. It is defined by:

$$f(out_{eval}) = \frac{I(out_{right}, out_{eval})}{H(out_{right})}, \tag{2}$$

where out_{eval} is the output vector of the evaluated network and out_{right} is the expected correct output vector for the problem. $I(X, Y)$ is the mutual information between X and Y, and $H(X)$ is the information entropy of X:

$$I(X,Y) = \sum_{x=0}^{1}\sum_{y=0}^{1} P_{XY}(x,y) \cdot log_2\left(\frac{P_{XY}(x,y)}{P_X(x) \cdot P_Y(y)}\right), \tag{3}$$

$$H(X) = \sum_{i=0}^{1} P_X(i) \cdot log_2(P_X(i)), \tag{4}$$

with $P_X(x)$ as the probability of $X = x$, and $P_{XY}(x, y)$ as the joint probability of $X = x$ and $Y = y$. This fitness function is defined in the range $[0, 1]$, with 1 as the best fitness. Due to using mutual information, both the correct output and its negation will have the same best fitness.

We conducted three sets of experiments: one using the cytoplasmic model, and two using the placental model with two different mD values (0.2 and 0.8). For understanding the role of maternal effects in network development, in each set we conducted experiments with dt fixed at 30 time steps, and used increasing values of mt. The used mt values correspond to periods of initial maternal influence for 0%, 10%, 30%, 50%, 70% and 100% of the total developmental time (mt/dt). Each case was conducted 10 times, with different random seeds in each run. The experiments were conducted with the ECJ (Evolutionary Computation in Java) package. Evolution was conducted for 300 generations with a population size of 600. Roulette wheel selection was used, and selected individuals for reproduction had a 50% chance of being mutated. The mutation operator, if it was applied, generated a new random value in only one slot (chosen at random) in the genome. The genomes used in the experiments contained 6 regulatory regions, with each region having 5 structural regions attached. The grid size was 5x5 units long, using $D = 0.06$ and $E = 0.1$ for diffusion. Fitness graphs for some typical runs are shown in figure 5 a). As it can be seen from the graph, evolution tends to occur with sudden jumps in fitness, but this can be justified by the fitness function alone, and it should not be related to the model itself. The currently

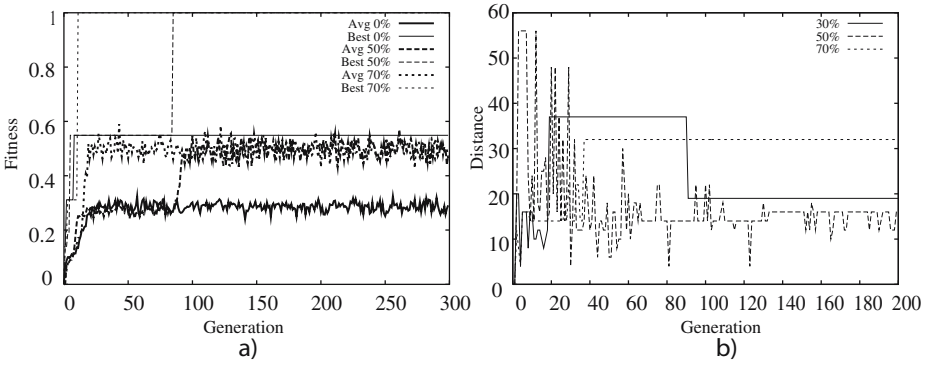


Fig. 5. a) Fitness graph b) Offspring sensitivity to maternal effects (both for the placental model with $mD = 0.2$)

used fitness function does not allow for a wide range of different values, therefore dynamics of this kind will always occur whenever this fitness function is used.

For further establishing the influence of maternal effects, we devised a simple distance measure between the neural networks. As the number and type of neurons are fixed for all the neural networks, we defined the distance between two networks as the total number of links that are unshared between them. Using this distance measure, we picked up the best individuals on each generation, grow them again without any maternal influence ($mt = 0$), and computed the distance between them. By doing this, we could therefore estimate the degree of sensitivity to maternal effects in each offspring. Typical results for a run are shown in figure 5 b). As it can be seen, offspring sensitivity tends to oscillate widely in the first generations, until eventually becoming stable. These oscillations, however, are only following dynamics in individual diversity, and on themselves do not show sensitivity to be adaptive. That is, diversity in offspring sensitivity in the earlier generations is only reflecting the larger diversity in the individuals in the initial population pool; as evolution progresses, the population diversity decreases, and therefore offspring sensitivity to maternal effects becomes stable.

For investigating any further influence in adaptiveness, we computed the average fitness over all generations, in all runs sharing the same mt parameter, with the results depicted in figure 6 a). Although this process may hide any special dynamics that may happen during evolution, at least it is able to show if there is any strong influence from the increasing mt values. All the three sets exhibit different effects on adaptiveness, with the two different models showing, in fact, opposite effects: the cytoplasmic model has an overall negative adaptive effect, while the placental model shows a positive one. For the $mD = 0.2$ experiments, maternal influence above 70% shows a roughly 30% increase in average fitness, a significant improvement ($p = 0.03$ with ANOVA). For $mD = 0.8$, however, this influence becomes stale, probably due to the mD value being too high.

In our opinion, there are three possible reasons for this improvement. Maternal influence may be positively affecting: 1) sensitivity to mutations, 2) develop-

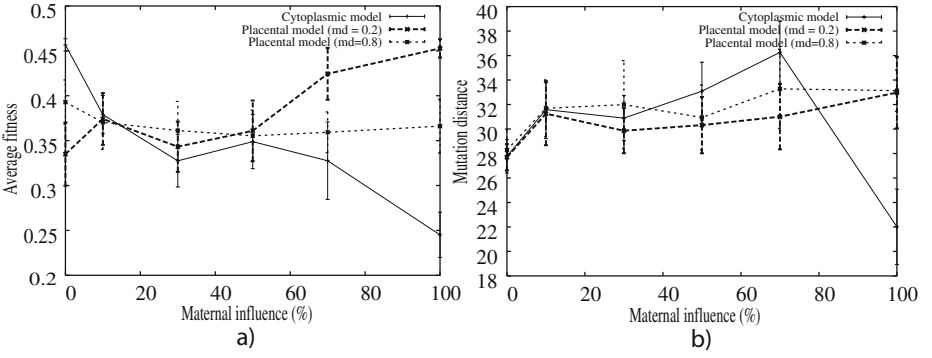


Fig. 6. a) Average fitness value for the runs, classified by increasing maternal influence. b) Robustness to mutations with increasing maternal influence. (average value + standard error)

ment, or 3) selection response. For checking the first hypothesis, we performed mutation experiments in the best individuals, and computed the average distance created by mutations. We picked up all the best individuals in all generations, mutated them once (generating children), and computed the distance between the original individual and its child. The procedure for mutation, and for computing the distance between the networks were the same as explained before. This process was repeated 10 times for each individual, and grouped by maternal influence. This, however, is not a good metric for computing mutation sensitivity, because development in the best individuals still has a strong influence from their own mothers. Therefore, we also performed mutation experiments where the individual was mutated twice in a serial fashion, yielding a grandchild; the distance computed was then between the original individual and its grandchild. Both experiments turned out to yield similar results, and the results for this latter case can be seen in figure 6 b). Maternal influence increased slightly the mutation sensitivity (except for the extreme case of 100%, in the cytoplasmic model), but this does not seems to be related to the adaptiveness increase. As for the second hypothesis, the placental model may be positively affecting development, by increasing diversity in the connections between neurons. In our model, redundant connections are ignored in the networks, and therefore increasing the number of connections can be a suitable strategy employed by evolution. Because the mother and offspring genotypes are different, they express different kinds of substances, that could increase link diversity in the offspring as the substances are exchanged. If the exchange rate is too high, however, it may prove too disruptive and therefore the effect is lost. We are currently checking this hypothesis, and also if delay in selection response occurs.

4 Conclusion

We presented two different models, reflecting two different mechanisms for maternal effects, using simulated development and a genetic algorithm. In our ex-

periments, the cytoplasmic model exhibited decreased adaptiveness in finding solutions, while the placental model showed significant adaptive improvements, especially with higher values of maternal influence, and with low exchange rates between mothers and offspring. This positive effect, however, was shown not to be related to any effect in mutation sensitivity. We are currently checking other possibilities for this effect, namely if maternal effects are influencing link diversity in the networks, or if they are delaying response to selection.

References

1. David E. Cowley. Genetic prenatal maternal effects on organ size in mice and their potential contribution to evolution. *Journal of Evolutionary Biology*, 4(3), May 1991.
2. Peter Eggenberger, Gabriel Gomez, and Rolf Pfeifer. Evolving the morphology of a neural network for controlling a foveating retina - and its test on a real robot. In Russell K. Standish, Mark A. Bedau, and Hussein A. Abbass, editors, *Proceedings of the Eighth International Conference on Artificial Life*, pages 243–251. MIT Press, 2002.
3. D. S. Falconer. Maternal effects and selection response. In S. J. Geerts, editor, *Genetics today, Proceedings of the XI international congress on genetics*, pages 763–774. Pergamon Press, Oxford, 1965.
4. Frederic Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Laboratoire de la Informatique du Parallelisme, Ecole Normale Superieure de Lyon, 1994.
5. Mark Kirkpatrick and Russell Lande. The evolution of maternal characters. *Evolution*, 43(3):485–503, May 1989.
6. MaryCarol Rossiter. Incidence and consequences of inherited environmental effects. *Annual Review of Ecology and Systematics*, 27:451–476, November 1996.
7. Jason B. Wolf, Edmund D. Brodie III, James M. Cheverud, Allen J. Moore, and Michael J. Wade. Evolutionary consequences of indirect genetic effects. *Trends in Ecology and Evolution*, 13(2):64–69, February 1998.

METAMorph: Experimenting with Genetic Regulatory Networks for Artificial Development

Finlay Stewart, Tim Taylor, and George Konidaris

Institute of Perception, Action and Behaviour,
University of Edinburgh
f.j.stewart@sms.ed.ac.uk, timt@inf.ed.ac.uk, gdk@cs.umass.edu
http://homepages.inf.ed.ac.uk/s0091983/METAMorph/

Abstract. We introduce METAMorph, an open source software platform for the experimental design of simulated cellular development processes using genomes encoded as genetic regulatory networks (GRNs). METAMorph allows researchers to design GRNs by hand and to visualise the resulting morphological growth process. As such, it is a tool to aid researchers in developing an understanding of the expressive properties of GRNs. We describe the software and present our preliminary observations in the form of techniques for realising some common structures.

1 Introduction

Genetic regulatory networks (GRNs) [6] have recently become a popular model of gene expression employed in genetic algorithms to study artificial developmental processes [2,3,1,4]. GRNs model the natural processes of intra- and inter-cell protein signalling during gene expression. However, the dynamics produced by GRNs and the properties of the associated morphological search space are difficult for researchers designing evolutionary systems to understand. The expressive properties of any encoding used as a basis for an evolutionary process impose a bias on the kinds of solutions that process is likely to find. It is therefore difficult to understand the dynamics of evolutionary GRN models without an understanding of the kinds of shapes that GRNs are biased toward expressing.

In this paper, we present METAMorph, an open source software application that allows for the hand-design and execution of artificial genomes that express cellular growth through GRNs. METAMorph aims to provide an environment within which we can experiment with one model of a GRN in order to understand its natural dynamics and the kinds of structures it is biased toward. It is also potentially valuable as a interactive teaching tool for understanding cellular growth processes. In the following sections we describe the GRN model used in METAMorph, provide a demonstration of its use in designing the development process of a cigar-shaped organism, and present a few design techniques for creating common structures.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 108–117, 2005.

2 Computational Development and Genetic Regulatory Networks

The field of computational development is the study of artificial models of embryonic cellular development, with the aim of understanding how complex structures and forms can develop from a small group of seed cells [4]. The natural way to study development is through the simulation of some abstract model of the dynamics of the cellular development process, often within the context of an evolutionary process.

GRNs model the interaction between genes, proteins, and the cellular environment. Each cell contains the same genome, but a potentially different set of active proteins, distributed across a fixed set of diffusion sites on the cell. Each gene has a set of enhancer and inhibitor proteins, which increase or decrease its activation, respectively, and when activated, produces some protein, using some output function (which scales its protein output according to its activation) and some distribution function (which places produced proteins at the cell's diffusion sites). Protein levels are also subject to attenuation (where protein levels decay) and diffusion (where proteins at diffusion sites move to other sites, in the same cell or on adjacent cells). Proteins can also be used as sensor mechanisms (where the cell produces them by itself under certain conditions), and as actuation mechanisms (where sufficient concentrations cause a cellular event, such as cell division, to occur). These processes are summarised in Figure 1. In this way, genes form a network whereby the interaction of the elements they produce regulates further gene expression. Variants of this basic abstract idea have been used in conjunction with an evolutionary process for a variety of applications,

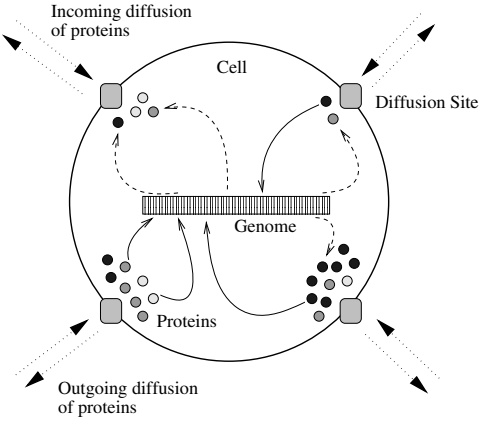


Fig. 1. Cellular GRNs. The same genome is expressed at all sites of all cells. However, local protein concentrations affect the production of new proteins at each site by enhancing and inhibiting genes. Proteins can diffuse between sites within a cell, and in some cases, between cells.

such as simulated cellular development [2,3,1,4], real-time robot control [5] and for the control of groups of underwater robots [7].

GRNs are thus a natural model for cellular development, and appear to possess desirable properties for acting as an evolutionary substrate [6] (e.g., they show a strong tendency towards modularity [1]). However, evolved GRNs are difficult for humans to understand, and we do not even have a qualitative measure of how difficult some natural structures are to achieve using them. In the following section, we introduce METAMorph, which aims to facilitate the accumulation of such knowledge through experimentation.

3 METAMorph

In the METAMorph (Model for Experimentation and Teaching in Artificial Morphogenesis) framework, multicellular artificial organisms are grown from a single cell (the *zygote*) using GRNs, with some subset of the proteins produced being able to trigger cell-level actions such as cell division or death. Many details are omitted for brevity; the interested reader is referred to the URL given above.

3.1 Proteins

A protein is defined by a unique name, a type (internal or external) and two constants: decay (τ) and diffusion (λ). Internal proteins may only diffuse within a cell, whereas external proteins pass through the cell membrane and can hence be used for inter-cellular signaling.

Internal Proteins. The proportion of the protein that is lost due to decay at each timestep is specified by τ:

$$conc_{prot}(t+1) = (1-\tau)conc_{prot}(t)$$

The concentration of each protein at 12 sites around the cell is stored; thus proteins may be unevenly distributed within the cytoplasm. The genome is expressed separately at each of these sub-cellular sites based on the local protein concentrations. The diffusion constant specifies the proportion of the protein that diffuses to neighbouring sites at each timestep. Due to the isospatial layout of the sites (see Figure 2), each one has exactly four equidistant neighbours between which this diffused protein is equally shared.

$$conc_{prot}(p,t+1) = (1-\lambda)conc_{prot}(p,t) + \sum_{q} neighbour(p,q)\frac{\lambda}{4}conc_{prot}(q,t)$$

External Proteins. Concentrations of external proteins are represented by isotropic 3D Gaussian distributions centred on the cell-site at which the protein originates. The diffusion constant specifies how much is added to the variance at each timestep. The decay constant determines how much the total concentration (i.e. the integral of the Gaussian function) should be reduced at each timestep. Note that external proteins diffuse freely through cells.

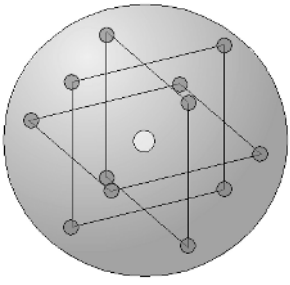


Fig. 2. The 12 protein sites (filled circles) are located at the corners of three mutually orthogonal squares centred on the centre of the cell (open circle)

3.2 Genes

All cells have the same genome comprising a number of genes. Each gene produces exactly one protein, although the same protein may be produced by several genes. The amount produced by the gene depends on zero or more promoter sequences attached to that gene. Each promoter sequence consists of a protein name and a weighting. Thus a weighted sum of protein concentrations at that site is calculated:

$$a_{prot} = \sum_{prot}^{promoterseqs} weight_{prot} conc_{prot}(x,t)$$

This value is the input for a sigmoid function with a certain bias and steepness defined in the gene:

$$conc_{prot}(x,t+1) = conc_{prot}(x,t) + \frac{1}{1+e^{-steepness(a_{prot}-bias)}}$$

3.3 Cells

Cells are represented as spheres of set radius. They each occupy a position on an isospatial grid in three dimensions (the same geometry as is used for the sites within cells), meaning that each cell can potentially have 12 equidistant neighbours. Cells can perform a number of actions, with an action being triggered when a specific protein's mean concentration in the cell exceeds a threshold value. These actions are as follows:

- **Cell division** When a cell divides it produces a daughter cell in the adjacent grid space in the direction of the mitotic spindle (see below), as long as that space is vacant. Cytoplasmic proteins may be shared unequally between mother and daughter, as the spatial distribution of proteins in the cell is taken into account during the split.

- **Mitotic spindle movement** Each cell has a 'mitotic spindle' that points in one of the 12 grid directions at any given time and defines the direction in which cell division takes place. This spindle may be moved forwards or backwards one step along an equatorial line around the cell as a result of a protein threshold being reached. Another action changes which 'orbit' the spindle is on. Alternatively, the spindle can be made to point in the direction of the sub-site where the concentration of a given protein is either highest or lowest.
- **Programmed cell death (apoptosis)** The cell is removed from the world leaving a vacant grid space.
- **Differentiation** Cells can have various different types. The type of a cell has no effect on its function, but is visualised by its colour. This feature is included to allow the investigation of how heterogeneous organisms can be created, e.g. in animals cells specialise as skin cells, blood cells, neurons, etc.

4 An Example Organism

This section describes how a cigar-shaped organism consisting of around 700 cells can be made. This is a fairly simple shape to create, as it is radially symmetric in all dimensions but one. It is included to give the reader a flavour of the techniques that can be used in morphogenetic design. A more detailed account (and downloadable demo) is available online, along with examples of more complex shapes.

The basic method of building the shape is as follows. A line of approximately 10 cells is created, which will form the central 'axis' of the cigar (Figure 3(a)). These cells all emit an external protein (Figure 3(b)). Cells will then grow wherever the concentration of this signal protein is above a certain level, which will result in a cigar shape being produced (Figure 3(c)).

To make cells proliferate in all directions, we can set up a genome where proteins that trigger division, spindle movements and orbit switches (let us call them *split*, *turn* and *switch*, respectively) have genes with thresholds ≤ 0, i.e. they will be expressed unless actively inhibited. An unchecked cell will therefore frequently move its spindle, switch the spindle's orbit, and divide. However, this behaviour must be prevented in some situations or we will just have an ever-growing ball of cells.

The cells comprising the axis are functionally different from the rest, as they must emit an external protein (call it *signal*). We can code this distinction by the expression of a protein: cells with a high concentration of *axis* belong to the axis. *Axis* can then enhance *signal*. It can also inhibit *turn*, since we require the axis to grow in a straight line. Finally, by making *axis* an enhancer for itself, we can lock cells in the axis state once the level of *axis* exceeds some threshold.

To limit the length of the axis, we ensure (by a process of trial and error) that the cells divide slightly more frequently (sharing their *axis*) than the autocatalytic nature of *axis* is able to replenish. Thus *axis* will become diluted with each division until it dips below the threshold needed to sustain itself. This cell

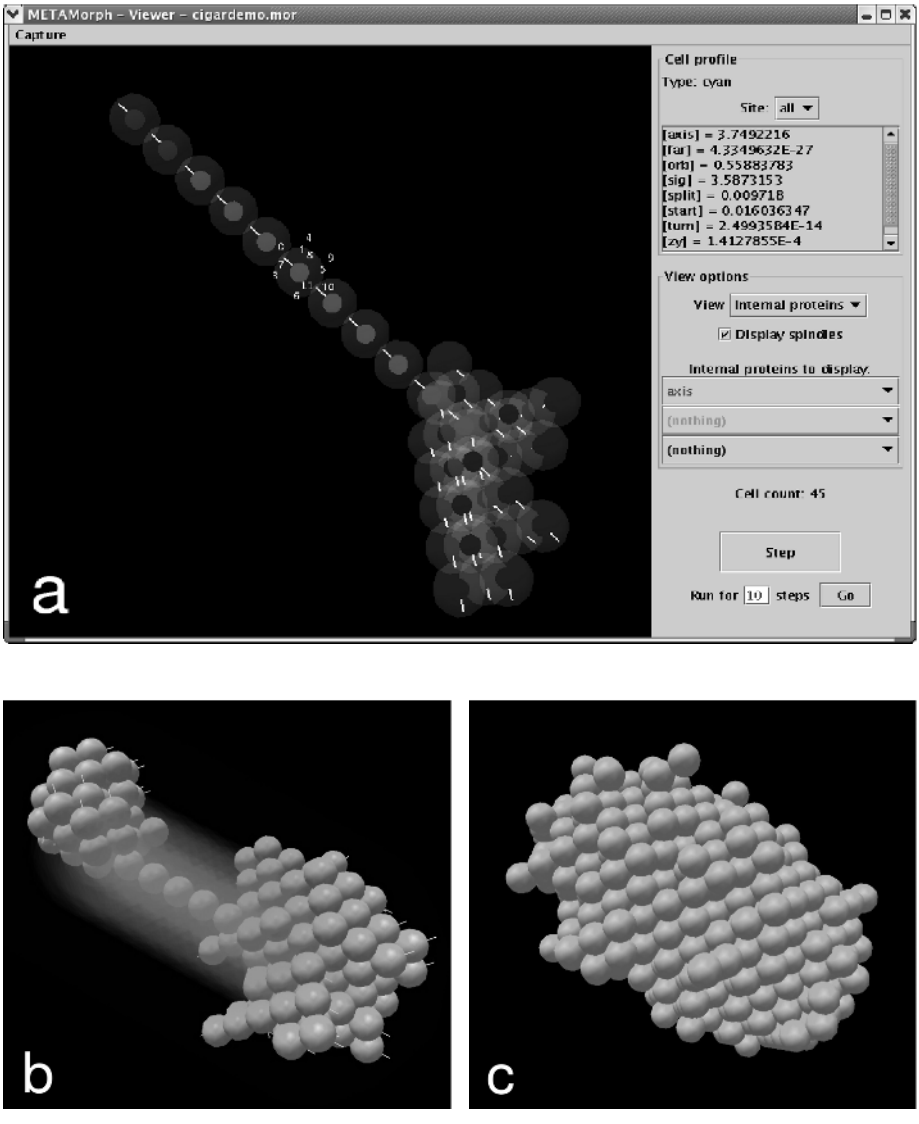


Fig. 3. Development of the cigar embryo. (a) Early in development, showing the METAMorph GUI. The light-centred cells are those with a high concentration of *axis*. The lines show the orientation of each cell's mitotic spindle; note that the axis cells' spindles are aligned as *axis* inhibits *turn*. Cells have started to proliferate from the zygote end of the axis (bottom right). (b) Cells are now growing from both ends. The cloud surrounding the axis represents the concentration of external protein *signal*. (c) The completed cigar, consisting of 684 cells. Note the existence of some offshoot cells which will soon die due to excessive levels of *far*.

will therefore become 'ordinary', i.e. it will not express *signal*, but will move its spindle, as will its daughters. In fact, by careful initial placing of proteins in the zygote, we can make the first cell become ordinary too, allowing the cigar to be 'filled out' from both ends.

All that remains to be explained is how we limit the outward growth of these cells to produce the desired shape. For this we need a protein (which we shall call *far*) that triggers apoptosis. It is inhibited by *signal*, so when the concentration of signal becomes too low (because the cell is far from the axis), *far* will accumulate and kill the cell. The organism therefore never reaches a stable configuration, but continually grows and kills cells at its periphery. Note that *axis* (or some other initially plentiful protein) must inhibit *far*, as there is no *signal* gradient set up at the beginning of the simulation.

One can now see how a few simple changes could be made to alter the eventual form of the organism. To make the cigar thicker, for example, one would simply set the threshold of the *far* gene higher, meaning that less *signal* would be needed to keep cells alive. Making the cigar longer is a little more complex, but could be achieved by setting the threshold of *axis*'s self-enhancing gene lower so that the cells of the spine could sustain more splits before the concentration of axis was too low to sustain the process.

To give an indication of the complexity of the algorithm, it requires 55 time-steps to reach a size of 680 cells, which takes approximately 7 minutes on a PC with a 2.4GHz Athlon processor.

While there is nothing very conceptually difficult about designing organisms in METAMorph, we have found that it typically involves a considerable amount of time-consuming trial and error. This is due to the inherently parallel nature of the morphogenetic processes. GRNs can suffer from a deleterious 'butterfly effect' whereby a seemingly innocuous change can have large unforeseen effects. For these reasons, designing a more complex 'quadruped' shape, consisting of a body with four appropriately sized and positioned limbs (see website), took one of us around a week.

Although our intention in this work was to investigate morphogentic processes by hand-designing organisms, METAMorph could be easily adapted to evolutionary experiments by adding code to generate candidate genomes and assess the resultant organisms according to some fitness function.

5 General Techniques

In this section we discuss a number of principles and techniques in morphogenetic 'programming', based on our preliminary experiments attempting to constuct various shapes. Some may exploit peculiarities in our model, but it is our hope that some may represent general mechanisms for morphogenesis, applicable to other multicellular GRN models, and perhaps even having analogues in biology.

Growth in all directions, with boundary control by external protein gradient. This technique for generation and maintenance of form is described

in section 4. It is robust due to its dynamic nature, as legitimate cells that are killed for any reason will grow back. Furthermore, the shape of the embryo can be easily altered *during a simulation* by changing the distribution of the signal protein.

Functional differentiation of cells. Since all cells share the same genome, it can be difficult to make some subgroup behave differently to the others. A good way to achieve this effect is to use a 'marker' protein (e.g. *axis* in the cigar example) whereby cells which have a non-negligible level of this protein will behave one way, the rest another. By making this protein autocatalytic (i.e. an enhancer for a gene producing itself) and using a substantially positive bias, this differentiation can be made to last indefinitely while still allowing cells to be switched either way by the use of other promoters. Note that this mechanism means that cell function will generally be inherited when a cell divides, which may or may not be desirable depending on the situation.

Delays. In some situations it can be useful for a cell to wait for a period of time before initiating an action. For instance, in creating a hollow shape, we may want to give cells a chance to grow outwards before killing off those that are too close to the centre. A way to achieve this effect is to create an 'accumulator' protein with a low decay constant. A fully-activated gene produces the accumulator for a number of time-steps, then another protein is produced when the accumulator finally reaches some threshold. Even longer delays can be created by using a chain of accumulators.

Quasi-binary processing. Protein concentrations and gene activations are continuous. While this is potentially useful, in many situations we require a sharp cut-off between states; e.g. a cell's membership of the cigar's 'axis' is a binary variable. A first step towards achieving this kind of behaviour is to use high steepness values for genes. Typically around 90% of the genes in our programs have step-like activation functions (steepness > 10). A typical situation is that where we have a continuously varying concentration of a 'primary' protein (often an external one), with a 'secondary' protein being produced in an all-or-nothing manner when its concentration is above or below a certain threshold level.

If we set the decay constant of this secondary protein high, its concentration at any time will depend chiefly on whether its gene was active on the last time-step (assuming that only one gene produces the protein). In this way, the presence or absence of that protein approximates a binary signal. If we have binary values, it is natural that we might want to perform simple logical operations upon them. This is not entirely straightforward. The problem lies in the fact that there is no well-defined concentration corresponding to '1' in protein logic. Any substantially non-zero concentration encodes '1', but the actual level depends on factors such as the decay constant, how recently the cell has divided, etc. It is therefore difficult to design, for example, an AND mechanism that will fire when both of its inputs are non-zero but not when just one of them is unusually high. (Note that devising an inclusive-OR mechanism is trivial.)

The key to solving the problem relies on the fact that there is a well defined '0' value, namely a concentration of 0.0. Thus, logical operations where all the clauses are negated can be carried out. It is straightforward to make a NOT mechanism by using a gene with a slightly negative bias, high steepness, and a large negative weighting on the input protein promoter. Hence, $P \wedge Q$ can be expressed as $\neg(\neg P \vee \neg Q)$. For example, when trying to pick out cells at a certain distance from a signalling cell (see below), we have found it useful to have a 'right distance' protein that is expressed when neither a 'too close' nor a 'too far' protein is present.

Location-specific cell selection. To build all but the simplest shapes, it is necessary to pick out cells to perform special functions according to their position in the embryo. A single cell can only produce a radially symmetric signal gradient, and hence any specific level of external protein is shared by a sphere of cell-sites. Uniquely pinpointing one location requires the intersection of four spheres, which requires four cells in different places emitting different signals. However, we have a chicken-and-egg problem: how can we position these cells, since they are required for positioning?

One solution we have found to be useful is as follows. Assuming one signal cell is already set up, the first cell which finds itself at the correct distance out can become the second signal cell, hence breaking the symmetry arbitrarily. Then, the first cell to find itself at appropriate distances from both existing signal cells can become the third, and so on. Of course, it is essential that once one cell has assumed a particular signalling role, none of the others do. This can be achieved if the presence of its signal inhibits other cells from emitting that signal. But this presents a problem: the concentration of the signal protein is obviously at its highest at the signalling cell. So how can we prevent it from inhibiting itself?

One method is to use an accumulator protein to create a delay (see above). Being at the correct distance from the existing signal cell(s) causes a certain protein to be produced. This protein triggers the production of the accumulator. Once the accumulator reaches a threshold level, an autocatalytic marker is produced, locking the cell into its role and causing the signal to be emitted. If, however, the signal is detected at any point during the waiting period, the process is immediately terminated. This mechanism ensures that only the first cell to end up in a suitable position will start signalling.

Once the multi-signal system is set up successfully, similar techniques can then be used for placing morphological features on the embryo. Returning to the cigar example, it would be possible to identify an outer cell and produce proteins which would lead to the production of *axis* and to the spindle being oriented in the direction of least *signal*. In this way, a 'limb' extending outwards from the central 'body' could be created (see the 'Quadruped' demo online).

In some cases, we may not want to globally inhibit other cells from responding when one does. For instance, we might want to set up two or more sites from which limbs will grow. In this situation we can alter the sensitivity to the signal such that only cells within a certain radius will be inhibited. Such lateral inhibition is useful for creating regularly-spaced structures, e.g. hairs on skin.

6 Summary

We have introduced and described METAMorph, a software platform for the experimental design of genomes encoded as genetic regulatory networks, given an example model of the development of a cigar-shaped body, and presented some of the design techniques that we have found useful. Although these are only a small step towards qualitatively characterising the expressive bias of the model of GRN used in METAMorph, we expect that further experimentation and open development will contribute to a more complete picture over time.

Acknowledgements

Facilities for this research were provided by the Institute of Perception, Action and Behaviour in the School of Informatics, University of Edinburgh. This work forms part of the HYDRA project (EU grant IST 2001 33060, `http://www.hydra-robot.com`).

References

1. J.C. Bongard. Evolving modular genetic regulatory networks. In *(Proc. IEEE 2002 Congress on Evolutionary Computation (CEC2002)*, pages 1872–1877. IEEE Press, 2002.
2. P. Eggenberger. Cell interactions as a control tool of developmental processes for evolutionary robotics. In *From Animals to Animats 4: Proc. 4th International Conference on the Simulation of Adaptive Behavior*, Cambridge, MA, 1996. MIT Press.
3. P. Eggenberger. Evolving Morphologies of Simulated 3D Organisms Based on Differential Gene Expression. In P. Husbands and I. Harvey, editors, *Proc. 4th European Conference on Artificial Life (ECAL97)*, Cambridge, MA, 1997. MIT Press.
4. S. Kumar and P.J. Bentley. An Introduction to Computational Development. In S. Kumar and P.J. Bentley, editors, *On Growth, Form and Computers*, chapter 1, pages 1–44. Elsevier, 2003.
5. T. Quick, C.L. Nehaniv, K. Dautenhahn, and G. Roberts. Evolving embodied genetic regulatory network-driven control systems. In W. Banzhaf, T. Christaller, P. Dittrich, J.T. Kim, and J. Ziegler, editors, *Proc. 7th European Conference on Artificial Life (ECAL2003)*. Springer Verlag, 2003.
6. T. Reil. Dynamics of gene expression in an artificial genome – implications for biology and artificial ontogeny. In D. Floreano, J.-D. Nicoud, and F. Mondada, editors, *Proc. 5th European Conference on Artificial Life (ECAL99)*, 1999.
7. T. Taylor. A genetic regulatory network-inspired real-time controller for a group of underwater robots. In *Proc. 8th International Conference on Intelligent Autonomous Systems (IAS-8)*, March 2004.

Morphological Plasticity: Environmentally Driven Morphogenesis

Katie Bentley and Chris Clack

UCL, Computer Science Department, London WC1E 6BT
k.bentley@cs.ucl.ac.uk,
www.cs.ucl.ac.uk/staff/k.bentley
c.clack@cs.ucl.ac.uk,
www.cs.ucl.ac.uk/staff/c.clack

Abstract. This paper focuses on the environmental role in morphogenesis in dynamic morphologies (DM). We discuss the benefits of morphological plasticity (MP) and introduce our Environment-Phenotype Map (E-P Map) framework in order to investigate and classify the *continual* development in DMs and morphologically adaptive behaviour. We present our MP-capable system the Artificial Cytoskeleton (ArtCyto), housed within our DM the 'Cellanimat', with an E-P Map closely based on MP examples from cell physiology. We provide experimental results to demonstrate that with this single E-P Map a bifurcation in morphology can occur, caused only by a difference in the environment, mirroring evidence from physiological data of fibroblast cell chemotaxis and macrophage cell phagocytosis.

1 Introduction

The central theme of 'New AI' [1], that embodiment is an integral part of intelligence, has resulted in increasing interest in system morphologies and their generation (morphogenesis). We classify morphologies into two groups based on the following criteria: if the morphological *structure*, by which we specifically mean the morphology's sub-component connectivity, can continually change in relation to the environment then it is a *dynamic* morphology (DM); otherwise it is a *static* morphology (SM). For example, a morphology grown with a developmental algorithm *before* entering the testing environment is classed as an SM (e.g. [2,3]); for the system's actual 'lifetime' (within the environment) there is no mechanism for the morphology's sub-component connectivity to change. Similarly, the linear-cluster robot 'morpho-functional machine' presented in [4] also classes as an SM with our definition, since its observably different 'straight form' and 'clustered form' are nevertheless structurally isomorphic.

DMs have morphological plasticity (MP). We can subdivide existing examples of DMs into two categories, based on MP concepts from physiology [5]. 1) irreversible: the DM can alter sub-component connectivity in relation to the environment; once set however, connections cannot be reversed (e.g. temperature-related sex determination in salmon). Investigations with L-systems in environments (e.g. [6] and those reviewed in [7]), are examples of irreversible MP in a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 118–127, 2005.

DM; once plant branches have grown, in accordance with environmental interactions, they cannot be changed. 2) Reversible: among DMs with *reversible* MP, where the same connectivity can be continually altered by environmental interaction, we count our Artificial Cytoskeleton (ArtCyto) model [8], other examples can be found in Artificial Chemistry (e.g. [9]).

MP allows organisms a greater chance of survival in fluctuating, changeable niches. A genotype capable of changing form by quick, lifetime responses to the environment has a selective advantage over one dependent on the slow process of evolution alone [10]. In a DM system, genetic information relating to morphology continually combines with the wealth of information in the environment, so with increased MP genotypes become more scalable, morphologies are kept relevant to current environmental conditions and adaptive behaviour can be morphological not solely controller-based.

A systematic measure of the DM environmental interface would not only provide a comprehensive framework for understanding and explaining morphogenesis mechanisms, but could help clarify, and indeed quantify, the level of embodiment of a system (as proposed by Quick in [11]). Therefore, in the next section we define our Environment-Phenotype Map (E-P Map) framework. We then present an example of environmentally driven morphogenesis using our DM model with reversible MP: the ArtCyto housed within the 'Cellanimat'. We provide experimental results to demonstrate that with a single E-P Map a bifurcation in morphology can occur, caused only by an environmental difference. We discuss the inter-relation of observed morphologies, behaviour and environment using our E-P Map framework. The mechanisms in this example are firmly based on the physiological mechanisms in fibroblast cell chemotaxis and macrophage cell phagocytosis.

1.1 MP Framework: The E-P Map

Here we introduce a general framework for understanding MP in DMs based on, and extended from, the open L-system approach for plant-environment interactions and classifications of morpho-functional machines (described in [7,12]). We introduce our term Environment-Phenotype Map (E-P Map) to describe environmentally driven morphogenesis (as appose to the genetically driven 'genotype-phenotype map') of a particular feature. An E-P Map is a set comprised of distinct environment phenotype interactions, or 'EP functions', whose competitive/cooperative combination within a specific environment can be used to explain observed morphological behaviours and structural changes.

We can define three possible types of EP function: 1) The morphological structure is affected by *global* environmental factors (e.g. gravity, temperature) 2) The morphological structure is affected by *local* environmental factors (e.g. obstacles, gradients) 3) The morphological-structural change has a *reciprocal* effect on the environment (e.g. depletion of environmental sources by uptake for growth). We further classify an EP function as either passive or active. Passive interactions do not cause or require the activation of *new* information-processing pathways in the DM. For example, a collision that simply blocks a DM's growth

(addition of connected sub-components), is a type 2-*passive* interaction, a collision that triggers sensor activation resulting in new sub-component behaviours and connectivity is a type 2-*active* interaction.

1.2 Chemotaxis and Phagocytosis

Fibroblast cell chemotaxis is an example of reversible MP. The cell detects a chemical gradient and transforms morphology in order to follow it to the source. The first stage of this requires the formation of a convex 'leading edge' with microspikes (protrusions) [13]. Phagocytosis, also reversible, is the process of engulfment of a foreign particle for degradation or ingestion; it is a fairly universal cell function also relying on profound rearrangements of the cell membrane. In macrophage cell phagocytosis (to ingest foreign particles), cell surface receptors trigger and bind to the particle, tethering it; this causes reactions involving the same proteins downstream as in fibroblast chemotaxis, but leads to a different morphology — in this case an enclosing concave morphology called the 'phagocytic cup' [14]. Chemotaxis (movement morphology) and phagocytosis (ingestion morphology) are distinct both topologically and functionally yet are controlled by the same underlying mechanism of *continual* environment-morphology interaction, thus the environmental difference causes the bifurcation in morphology.

2 The Cellanimat Model

The Cellanimat is our DM; morphology is determined at each timestep by an E-P Map for reversible MP. The Cellanimat differs from the traditional animat approach as it has no separate controller (processor) from the body (effector): instead the body acts as a combined processor/effector. The Cellanimat is a dynamical hierarchy, modelling real cells at three levels: 1) the cell and its environment; 2) the cell's subsystem interaction (membrane, transduction pathway, cytoskeleton, cytoplasm); 3) the subsystem's macromolecular interactions.

The key subsystem for structural change in all real, eukaryotic cells is the cytoskeleton, a complex, dynamic network of protein filaments which extends throughout the cytoplasm. In particular *actin* cytoskeleton microfilaments are involved in rapid changes to membrane shape in response to environmental signals [13]. We presented the ArtCyto in [8], designed to model these dynamics. The ArtCyto is the core processor/effector of the Cellanimat. It consists of *structural* proteins modelled as agents (actin and a nucleator), which make up the filaments, and *accessory* proteins, which regulate filaments' behaviour (e.g. inhibiting, activating, severing, bundling) [13]. Accessory and structural protein behaviour and connectivity, can be controlled by environmental signals. Signals filter into the cell via the *transduction pathway* (TP), a cascade of reactions triggered by cell-surface-receptor activation (sensors), dramatically altering filamentous structure. This in turn affects the shape and structure of the cell as a whole. See Fig.1(a) for a schematic of the Cellanimat.

The Cellanimat and its environment are implemented as a 3D voxellated world. It is an idealized model of Nature (partially inspired by artificial chemistry

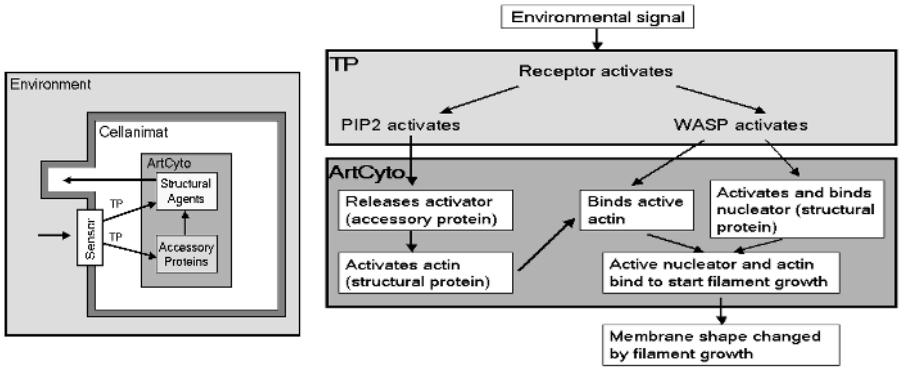


Fig. 1. (a) Schematic of the Cellanimat with environmentally driven membrane shape change (b) EP_1 from the E-P Map as in [8], abstracted from the biological pathway for fibroblast chemotaxis

techniques [15]) that concentrates on the computational process of adaptation; the mechanisms of macromolecular interaction are faithful to a subset of those in fibroblast cells, but it is not an exact model of a cell. The voxels can be in one of three states: environment (outside the cell), cytoplasm (inside the cell), and membrane (the cell border). In the Cellanimat, the TP is nested within the membrane and the ArtCyto within the cytoplasm. Changes in morphology are evident from the distribution of membrane voxels. The ArtCyto structural agents have knowledge of their position, binding sites and time spent in the current filament. All other Cellanimat components are modelled within a cellular automata. All macromolecules (agents or CA) exist in single voxels and interact with their 26 nearest neighbours (NNs) according to given rules. Thus all voxels have sub-states, indicating the presence and properties of macromolecules, and where appropriate their concentration (which may diffuse to neighbouring voxels, calculated using the method in [16]).

2.1 The Protrusions E-P Map

The experiments in this paper use the E-P Map described more fully (e.g. with exact local rules) in [8], which closely models the underlying mechanism controlling MP in fibroblast chemotaxis and macrophage phagocytosis. The protein selection was as follows: environment voxels may contain a concentration of chemoattractant (C) or denote a foreign particle (P); membrane voxels may contain a receptor, WASP and/or PIP2 (TP proteins); and cytoplasm voxels may contain, from the ArtCyto, either a structural agent (actin/nucleator), or a concentration of the accessory protein 'activator'. The E-P Map can be subdivided into the following three EP functions.

EP_1: 'filament formation', **type 2-*active***. For the protein state change rules upon receptor activation by C or P, see See Fig. 1(b) and [8] for full details.

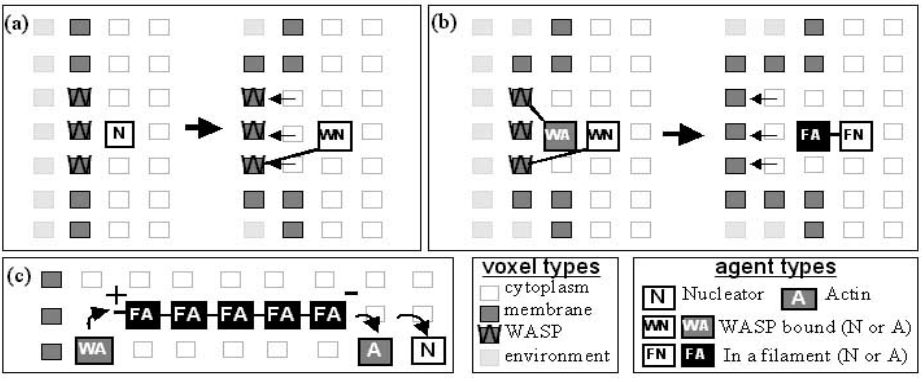


Fig. 2. Diagrams of EP_1 rules: (a) inactive nucleator (N) binds to a WASP neighbour becoming active W-N 'pushing out' the membrane; (b) W-actin and W-N bind, starting a filament. Local membrane is 'pushed out'; (c) W-actin continues to bind to a filament, 'pushing out' membrane, as F-N and F-actin fall off and deactivate at the other end.

Actin agents cycle through a succession of four states: inactive, active (bound to activator), active and bound to WASP (W-actin), active and bound to a filament (F-actin). The rules for nucleator agents cause them to cycle through: inactive, active and bound to WASP (W-nucleator) and active and bound to a filament (F-nucleator). The default agent state is inactive with random movement. A W-nucleator starts a fresh filament by binding W-actin, Fig. 2(b). Thereafter, filaments grow by accruing W-actin, even if the nucleator is lost. Filaments may grow in any direction, but as WASP-bound agents (W-agents) are needed, filaments grow preferentially towards the cell membrane (containing WASP).

Over time, agents disassociate from the nucleated end of a filament (the filament continues to grow at the opposite end) and may then be recycled for use in the same or another filament, Fig. 2(c). Filaments appear to move towards the membrane due to this process. When an agent in a filament has membrane in its NNs, it replaces these with cytoplasm voxels and reassigns all current exposed environment voxels to membrane state, thus it has 'pushed-out' the membrane locally, Figs 2 (a),(b). To conserve cytoplasm volume, membrane voxels are simultaneously pulled in at the furthest point in the cell having no adjacent filaments (a simple model of surface tension effects).

EP_2: 'collision', **type 2-*passive***. Local membrane 'push out' is blocked by solid environment voxels (P or environment boundary) in its NNs, blocking addition of agents to that filament. EP_3: 'redistribution', **type 3-*passive***. Any concentration of C in the environment voxels that 'push-out' overwrites are redistributed to their NNs first (conserving C volume).

In each timestep the Cellanimat first performs a **CA-synchronous update**: 1) C in the environment is diffused; 2) membrane voxel TP proteins are activated or deactivated, activated PIP_2 releases activator into a cytoplasm NN; 3) activator is diffused. Then the Cellanimat performs, in sequence, **agent-**

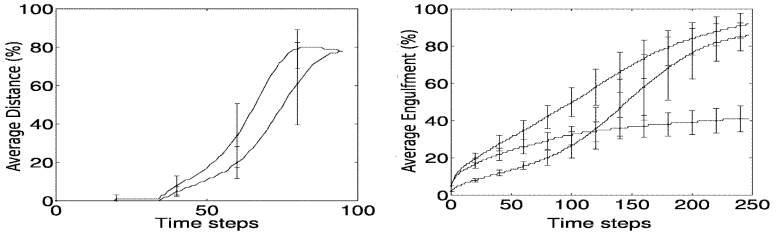
asynchronous update: 1) agent connectivity is updated based on NNs (e.g. nucleator becomes W-nucleator if it has a WASP NN (Fig. 2(a)); 2) if not in a filament, agents perform random movement to an NN; W-agent movement is constrained to ensure WASP NNs; 3) local membrane and environmental adjustments are made as necessary.

3 Experiments and Results

We aimed to test the multifunctionality (in behaviour and morphology) of the protrusions E-P Map. Our primary investigation therefore used the E-P Map for chemotaxis and applied it to a phagocytosis problem, by replacing only the environment (C with P). We also performed two further investigations: (i) improving the E-P Map and (ii) exploring the limits of phagocytic cup morphology. All results were averaged over 100 runs. For all experiments the Cellanimat was cylindrical (radius 20 voxels, height 6 voxels). Agents occupied 75% of the cytoplasm volume, 35% of which were nucleators, all initiated at random positions (there were 4,924 structural agents in total). Receptors occupied 20% of membrane voxels. See Table 1 for a breakdown of all results.

Table 1. Averaged behavioural results (distance or engulfment); Chemo t=100, Phago t=250

Experiment	Description	Mean dist/engulf	Standard deviation
Exp 1	Chemo	78%	12%
Exp 2	Phago	41%	7%
Exp 3	Chemo WASP r=2	78%	3%
Exp 4	Phago WASP r=2	92%	6%
Exp 5	Phago WASP r=2 large particle	86%	7%



(a) (b)

Fig. 3. (a) Average distance covered at each timestep (%) over 100 runs of chemotaxis: Exp 1 (lower) Exp 3 (upper) (b) Average engulfment at each timestep (%) over 100 runs of phagocytosis: Exp 2 (lower) Exp 4 (upper) Exp 5 (middle)

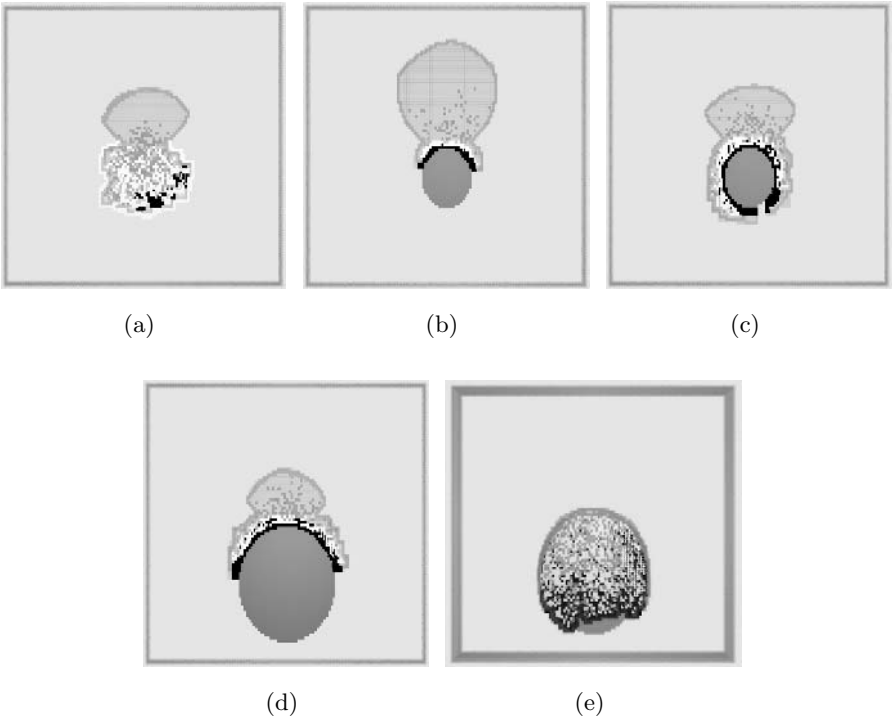
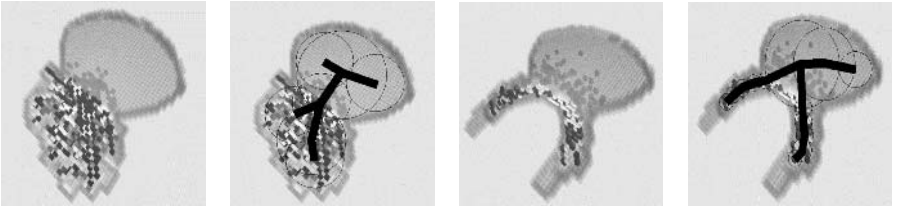


Fig. 4. WASP (Black); filaments (white); active actin (grey); membrane(grey). slice view of first 3 layers only (a) Exp 1: chemotactic leading edge, white voxels outside cell contain C, timestep t=90 (b) Exp 2: Phagocytic cup t=250 (c) Exp 4: t=250 (d) Exp 5: t=125 (e) Exp 5: full top view t=250.

3.1 Multifunctionality

The Cellanimat was first tested at chemotaxis (Exp 1). A plume of diffusing C was dropped 30 voxels away from the Cellanimat's centre and the distance covered by the Cellanimat to reach the plume was calculated as its 'movement behaviour'. Fig. 3(a) shows that by 100 timesteps (t=100) the Cellanimat had, on average, moved its centre of mass 78% of the distance towards the C plume. The leading edge morphology is shown in Fig. 4(a). The morphology returned to a cylindrical form and further distance was not covered due to EP_3 being of type 3; redistribution caused the Cellanimat to become completely submerged in C triggering EP_1 in all directions, evening out the form and inhibiting movement.

The Cellanimat with the same E-P Map was tested at phagocytosis, the 'engulfment behaviour' was measured as the number of available particle voxels with membrane NNs. A particle with radius 10 voxels and height 3 voxels was initiated, tethered to the Cellanimat, on the floor of the environment with its centre of mass at the location of the initial plume drop site in Exp 1. Cup



(a) (b) (c) (d)

Fig. 5. The two morphologies (Exp 3, Exp 4) and their medial axis functions (environment omitted)

morphology is visible in Fig. 4(b). We can see from Fig. 3(b) that the behaviour reaches a plateau, resulting from the competition of EP_1 and EP_2; particle P both activates and inhibits growth.

Morphometrics. A clear difference in observable morphology is evident from Figs. 5(a) and 5(c), the chemotaxis morphology (Morph 1) is convex, whereas the phagocytosis morphology (Morph 2) is concave. In a convex morphology advancing membrane voxels assist the advance of neighbouring membrane voxels. By contrast in the concave Morph 2, leading membrane voxels are prevented from moving ahead by the particle itself (EP_2). A cup morphology is thus intrinsically more difficult to achieve. The difference can be quantified by calculating a Medial Axis Function (MAF) [17] for both morphologies; see Figs. 5(b) and 5(d). In Morph 1, the illustrated medial axis contains a middle "body" section, whereas the Morph 2 medial axis has none. Futhermore, the radius functions of circles forming the medial axis "legs" in Morph 1 are four times Morph 2's. A further difference results from EP_3 in Exp 1, C is eventually displaced so Morph 1 starts to grow haphazardly, whereas Morph 2 remains smooth and controlled.

3.2 Improving the E-P Map

In order to allow filaments to form just to the side of the particle, and thus avoid inhibition by EP_2 we hypothesized that an increase in the radius of WASP activation, upon receptor activation, would increase the viable range for filament formation, as W-agents are needed for filaments. We therefore re-tested the Cellanimat, with WASP activated in all membrane voxels within a radius of 2 voxels (rather than 1) from an activated receptor, in Exp 3 (chemotaxis) and Exp 4 (phagocytosis). See Table 1, increasing the WASP radius had little effect on chemotaxis as EP_2 inhibition was never a problem, but it greatly increased engulfment in Exp 4, see Fig. 4(c) for full cup morphology.

Wider WASP recruitment in Exp 4 increased the range of viable locations for EP_1 (as EP_2 inhibits growth into the particle). However, engulfment ability in real cells is more likely due to two factors: 1) the property of nucleators to stick to an already formed filament, starting a branch and 2) the flexibility of filaments.

Flexibility and further branching would allow filaments to grow around the side of the particle, rather than stubbornly trying to grow in straight lines into it and failing. Branching is possible in the Cellanimat, but nucleators deactivate when disassociated from WASP, thus branching is actually very rare. It is not known whether nucleators stay active after WASP disassociation in real cells; this would seem worthy of further investigation. Increased WASP radius was a simple way to fully achieve engulfment with the model as it stands, but deeper reworking of the model may be useful.

3.3 Extreme Ingestion

In the paper 'How to Eat Something Bigger Than Your Head', Aderem showed that a cell can engulf a particle larger than itself through the recycling of internal membranes [18]. In Exp 5 we tested our improved system against a particle equal in radius to the Cellanimat, as the membrane can stretch indefinitely in our model. See Table 1 and Figs. 4(d) and 4(e); the Cellanimat stretched its entirety around the particle engulfing an astonishing 86% on average.

4 Summary and Conclusions

We have argued that, in a DM, morphogenesis is an adaptive behaviour. We have discussed benefits and mechanisms of morphological plasticity (MP) and introduced our E-P Map framework in order to formalize and understand morphologically adaptive behaviour. We presented our DM model the Cellanimat, and its core processor/effector the ArtCyto, closely based on eukaryotic cell physiology. We demonstrated its effectiveness with experimental results to show that with a single E-P Map a bifurcation in morphology, and behaviour, can occur, caused only by a difference in the environment.

We believe that the Cellanimat and E-P Map approach leads to improved understanding of morphogenesis and adaptive behaviour; our Cellanimat currently models a subset of cell physiology, we are exploring which further aspects of biological fidelity are useful for achieving more complex adaptive behaviour.

Acknowledgements

Thanks to A. Cameron (Cancer Research UK), A. Koffer (UCL Physiology) and E. J. Cox (Natural History Museum) for patient, invaluable biological insights.

References

1. Brooks, R. A.: Intelligence without reason. In Proc. IJCAI-91. (1991) 569-595
2. Bongard, J.: Repeated Structure and Dissociation of Genotypic and Phenotypic Complexity in Artificial Ontogeny. In: Spector, L. and Goodman, E. D. (eds.): GECCO-2001. Morgan Kaufmann (2001) 829–836

3. Hornby, G., Pollack, J.: The advantages of generative grammatical encodings for physical design. In Congr. on Evolutionary Computation (2001) 600–607
4. Kawai, N., Hara, F.: Formation of Morphology and Morpho-Function in a Linear Cluster Robotic System. In: Pfeifer, R. et al (eds.): From Animals to Animats 5, Proc. 5th Int. Conf. SAB'98 (1998) 459–464
5. Piersma, T., Drent, J.: Phenotypic Flexibility and the Evolution of Organismal Design. TRENDS in Ecology and Evolution **18**(5) (2003) 228–233
6. Rust, A. G., Adams, R., Bolouri, H.: Evolutionary Neural Topiary: Growing and Sculpting Artificial Neurons to Order. In: Bedau, M. et. al. (eds.): Proc. of ALIFEVII (2000) 146–150
7. Mech, R., Prusinkiewicz, P.: Visual Models of Plants Interacting with their Environment. Proc. 23rd Ann. Conf. on Computer Graphics SIGGRAPH (1996) 397–410
8. Bentley, K., Clack, C.: The Artificial Cytoskeleton For Lifetime Adaptation of Morphology. In: Bedau, M. et al (eds.): Workshop Proc. of the 9th Int. Conf. on the Simulation and Synthesis of Living Systems (2004) 13–16
9. Suzuki, K., Ikegami, T.: Self-repairing and Mobility of a Simple Cell Model. In: Pollack, J. et. al. (eds.): Proc. 9th Int. Conf. on Synthesis and Simulation of Living Systems (2004) 421–426
10. Behera, N., Nanjundiah, V.: Phenotypic Plasticity can Potentiate Rapid Evolutionary Change. J. Theo. Biol. **226** (2004) 177–184
11. Quick, T., Dautenhahn, K., Nehaniv, C., Roberts, G.: On Bots and Bacteria: Ontology Independent Embodiment. In: Floreano, D. et. al. (eds.): Proc. of ECAL99 (1999) 339–343
12. Pfeifer, R.: Morpho-functional Machines: Basics and Research Issues. In: Hara, F. Pfeifer, R. (eds.) Morpho-functional Machines: The New Species. Springer-Verlag (2003) 1–21
13. Alberts, B.,Bray, D.,Lewis, J.,Raff, M., Roberts, K., Watson, J. D.: Molecular Biology of The Cell. 3rd Edition. Garland Publishing (1994)
14. Castellano, F., Chavrier, P., Caron, E.: Actin Dynamics During Phagocytosis. Seminars in Immunology **13** (2001) 347–355
15. Hutton, T. J.: Evolvable Self-Replicating Molecules in an Artificial Chemistry. Artificial Life **8**(4) (2002) 341–356
16. Glazier, J. A., Graner, F.: Simulation of the differential adhesion driven rearrangement of biological cells. Physical Review E **47**(3) (1993) 2128–2154
17. Lee, D. T.: Medial axis transformation of a planar shape. IEEE Trans. Pattern Anal. Mach. Intell. PAMI. **4** (1982) 363–369
18. Aderem, A.: How to Eat Something Bigger Than Your Head. Cell **114**(1) (2002) 5–8

A Self-organising, Self-adaptable Cellular System

Lucien Epiney[1] and Mariusz Nowostawski[2]

[1] Swiss Federal Institute of Technology,
(EPFL) Lausanne, Switzerland
lucien.epiney@epfl.ch
[2] Department Information Science,
University of Otago, Dunedin, New Zealand
mariusz@nowostawski.org

Abstract. Inspired by the recent advances in evolutionary biology, we have developed a self-organising, self-adaptable cellular system for multitask learning. The main aim of our project is to design and prototype a framework that facilitates building complex software systems in an automated and autonomous fashion. The current implementation consists of specialised programs that call (co-operate with) their local neighbours. The relationships between programs self-assemble in a symbiotic-like fashion.

Specialisation is achieved by stochastic exploration of alternative configurations and program space. A collection of global and local behaviours have been observed and investigated. Based on preliminary experimental results, certain behaviours that spontaneously exhibit self-organisation and self-assembly are discussed.

1 Motivation: Multitask Learning

The artificial life system presented in this paper aims at solving multiple problems. It is especially efficient in *multitask learning*. Multitask learning is an area of machine learning which studies methods that can take advantage of previously learned knowledge by generalising and reusing it while solving a set of possibly related tasks [2]. Multitask learning has already shown promising results when applied to Artificial Neural Networks [4].

Some argue that multitask problems that share similar internal structure are much more common than one could imagine [16]. Most likely this maybe a certain feature of all living systems. The human being, for instance, is continuously confronted with new tasks. Human has to solve these tasks in parallel, using a single brain that has accumulated experience about all the previous tasks encountered since the birth. On another level, populations work in similar way. In computer science, traditionally multiple problems are translated into single task problems. The main advantage of multitask learning is the ability to reuse the previous acquired knowledge. Solutions (or parts of them) of previous problems, can be reused to solve the current tasks. The system can shift its bias to search for a hypothesis space that contains good solutions to many of the problems in the environment.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 128–137, 2005.

Requirements. In order to be efficient in a multitask context, our model should fulfill the following requirements:

1. All tasks must be solved (eventually).
2. Solving difficult tasks should lead to greater rewards than solving easy ones.
3. Computational resources should focus on unsolved tasks.
4. Solutions must not be forgotten, as long as they are useful.
5. Knowledge diffusion should be facilitated (previous solutions must be accessible).
6. Dynamic environments should be supported: tasks can be added and/or removed at any time, dynamically.

Experiments. For preliminary testing, we used a set of simple arithmetical tasks like $2x$, $|x|$, $3x + 2y$, $49 - x$, etc. These tasks enable the creation of related, incremental problems, that will highlight the main features of our model. What's more, it is straightforward to tune the desired difficulty level according to the computational resources at disposition.

2 Biological Inspirations

Symbiosis is defined as the interaction between two organisms living together. At least one member benefits from the relationship. The other member (the host) may be positively or negatively affected. Proponents of symbiogenesis argue that symbiosis is a primary source of biological variation, and that acquisition and accumulation of random mutations alone are not sufficient to develop high levels of complexity [6, 7].

K. Mereschkowsky [9] and I. Wallin [18] were the first to propose that independent organisms merge (spontaneously) to form composites (new cell organelles, new organs, species, etc). For example, important organelles, such as plastid or mitochondria, are thought to have evolved from an endosymbiosis between a Gram-negative bacterium and a pre-eukaryotic cell. A similar hypothesis can also be made regarding the origin of the nucleus [5]. According to Margulis [8], "Life did not take over the globe by combat, but by networking".

Another phenomenon widely spread in nature, that occurs at all levels of biological organisation from molecules to populations, is *specialisation*. As an example, the cells of a vertebrate body exhibit more than 200 different modes of specialisation [1]. Specialisation is the process of setting apart a particular sub-system (reducing its complexity) for better efficiency of a particular function. Our working hypothesis is, that specialisation together with symbiosis is necessary to reach higher complexity levels.

Recent work in incremental reinforcement learning methods also advocate retention of learnt structures (or learnt information) [12]. The sub-structures developed or acquired during the history of the program self-improvement process are kept in the program data-structures. It is therefore surprising that this general procedure is not being exhibited by any of the (standard) evolutionary programming models, such as

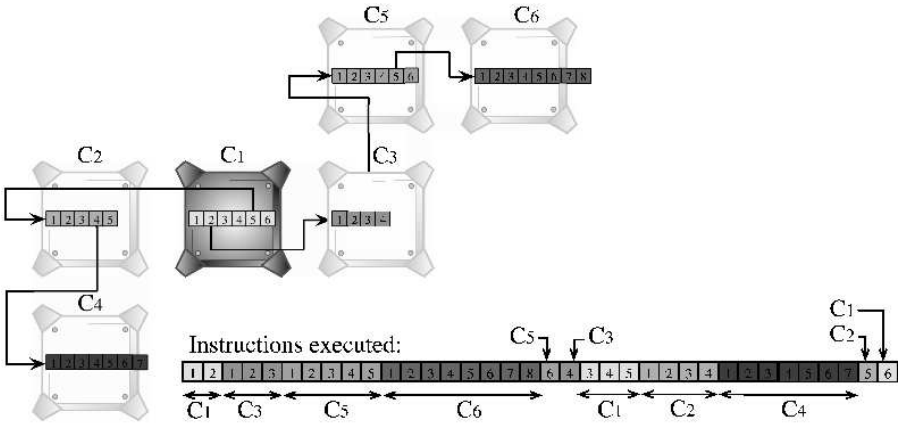


Fig. 1. The dark cell executes its program. Arrows show instructions that call neighbours' programs.

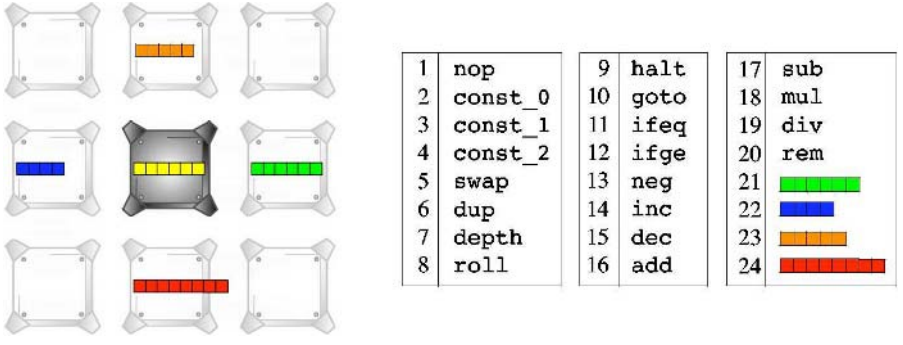


Fig. 2. Instruction set I for the dark cell's program. Instructions 21 to 24 execute programs of its neighbours. The von Neumann neighbourhood is used for this example. A program can thus look like the following: `add dup` *leftNeighbourProgram* `mul` *rightNeighbourProgram*.

GP or GAs [17]. Although these evolutionary programming models are inspired by biological evolution, they do not share some significant aspects that are recognised in current evolutionary biology, neither can they be used (directly) in an incremental self-improvement fashion[1].

[1] GA and GP maintain some developed substructures during the course of evolution towards a particular solution, however, as soon as the equilibrium is reached, or the optimal solution found, all the intermediate substructures are quickly "forgotten". When trying to learn a new task, the search process must start from scratch again, and the common practice in the application of GA/GP is to restart the search from a random population. In contrast, self-improvement in multitask environment never restarts the search from a random population for new tasks. The idea is to maintain and reuse previously evolved structures.

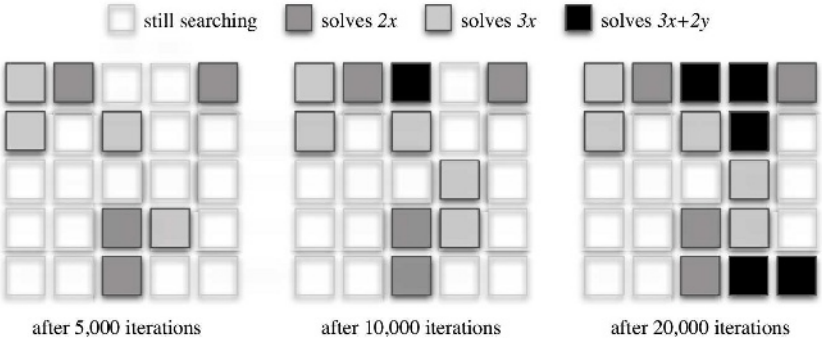


Fig. 3. Typical run exhibiting self assembly. 3 tasks: $2x$, $3x$, and $3x + 2y$. After 5,000 iterations, several cells can solve the two simple tasks ($2x$ and $3x$). After 10,000 iterations, one cell ($C_{1,3}$) uses its left and down neighbours to solve the hard task ($3x + 2y$), and all these 3 cells share the rewards (symbiosis). Shortly afterwards, some of its neighbours take advantage of it: they simply solve the task by calling $C_{1,3}$ (parasitism). Finally, after 20,000 iterations, we can observe another cluster of solutions at the bottom of the grid.

3 A Self-organising, Self-adapting Cellular System

Cellular systems are characterised by many locally interacting components (cells), therefore performing a parallel, decentralised, highly redundant computation [13]. In biology, cellular systems such as insect colonies, cellular tissues, or brain, have proved to be efficient, adaptive, and robust. Because of the absence of global, central control, they are not prone to major failure. Their emergent properties have been successfully applied in diverse fields of computer science, e.g. Cellular Automata [19], Membrane Computing [14], or Ants Algorithms [3].

A cell is the fundamental component of our system. All cells run asynchronously and in parallel. They react to local interactions with their neighbours and with the environment. Every cell maintains a program[2] P. The cell's goal is to find a successful program: one that, by solving a task, yields enough rewards (food) to survive. Programs can call neighbours' programs (Figure 1 and Figure 2). Moreover, if a program gets a reward, it will share it with any neighbour's program used to compute the solution. Both of them will benefit from their relationship. In other words, symbiotic relationships may appear between programs. This ability to access neighbours' programs has thus opened the door to complex hierarchical organisation (self-assembly). As a consequence, cells are now able to collaborate to solve complex problems (Figure 3).

Topology. There is no particular constraint on the topology of the web of cells. It can be n-dimensional and the neighbourhood can be as big as desired. In the extreme case, we could imagine a network in which every cell can access every other cell. Even dynamic topologies, that change in time, are possible. For our experiments, we use a

[2] A program is a sequence of instructions. For our implementation, we used a stack-based assembly-like programming language with universal computation capabilities [10].

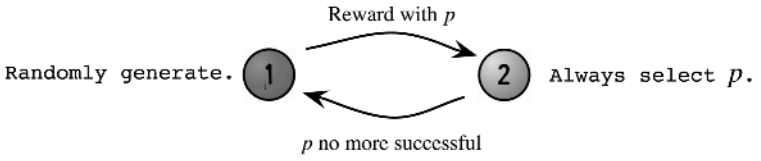


Fig. 4. The **random search mechanism** has two states. In the first, initial state, it randomly generates a new program for P. If a selected program p is rewarded, the mechanism transits to the second state. In the second state, p is always selected for P. It will stay ('survive') in that state as long as p remains successful. It is implemented by storing the cumulative rewards (called $provisions$) gained by p. At every time step, a fixed amount is subtracted from $provisions$. If the cumulative rewards drops below zero, p is considered unsuccessful and the mechanism transits back to state 1 (the cell dies).

toroidal grid of cells with a von Neumann neighbourhood (4 neighbours: right, left, up, and down).

Cell specialisation consists in finding a successful program for the cell. In other words, the cell self-adapts to a particular task in the environment. Several specialisation mechanisms have been studied by the authors: classic genetic algorithm, ad-hoc stochastic search maintaining a tree of probabilities of potential building blocks, or an adaptation of an environment-independent reinforcement learning method proposed by Schmidhuber [11]. However, since this paper focuses on the global behaviour of our system, we will present only the results obtained with the simplest specialisation mechanism: random search (Figure 4).

4 Environment

The environment represents the external constraints on our system (Figure 5). Its role is to keep the system under pressure to force it to solve the tasks. We have designed the environment as a set of *resources*. There is a one-to-one mapping between the resources and the tasks to solve (every resource corresponds to a task). The purpose of these resources is to give rewards to the cells when they solve their task. How many rewards are given and how accessible are the resources is the topic of this section. Every resource has two attributes: *quantity* and *quality*. Values for these attributes specify how much food (reward) will be given to the cell that consumes the given resource.

Resource's quantity. This parameter ($QUANTITY$, capitalized to highlight its static nature) represents the abundance of resources in the environment. This value is set ab initio and is the same for each of the resources. It allows us to tune the amount of cells that will be able to survive.

Resource's quality. The resource's quality has to reflect how difficult a task is. It facilitates a mechanism to give more rewards for hard tasks (requirement 2) as hard tasks may involve several cells that have to share the rewards. There are several ways of measuring the difficulty of a task. Some are *ab initio* (using expert knowledge), but it is more interesting to adjust it dynamically based on the observed difficulty. For example, the resource's quality may be set based on the observed average time it takes to solve

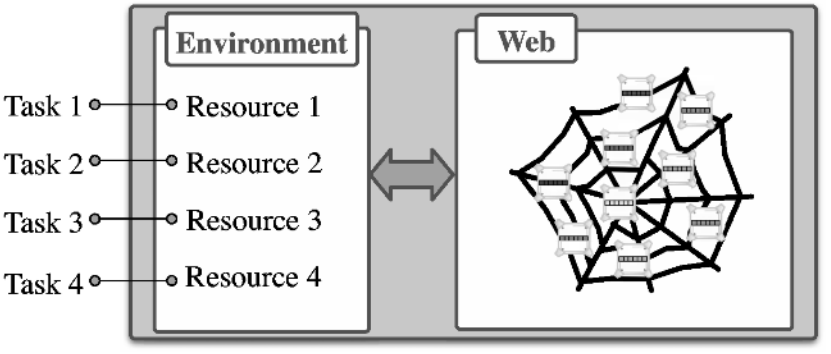


Fig. 5. The global view of our system. A dynamic set of tasks is provided to the system (requirement 6). Tasks are mapped to environmental resources. A web of cells interacts with the environment trying to solve the tasks. We use a schematic representation for the web to stress its flexible connections.

it, or on how many cells can solve it, etc. We decided to set the resource's quality to the current minimal number of cells required to solve the task. It will reflect dynamically the task's complexity without depending on randomness and without the use of extra parameter that would need to be tuned for the search process.

Food. When a cell consumes a resource, it gets the following amount of food (rewards):

$$food = \frac{QUANTITY \cdot quality}{consumers}\,, \tag{1}$$

where $consumers$ is the number of cells consuming the resource. Moreover, a cell has to share its food with all the neighbours it used to solve the task. Every cell used will get the same share of food.[3]

At every iteration, a cell needs to eat one unit of food. If it eats more, it can makes provisions by storing it. On the other hand, if it eats less it will die from starvation once its provisions are empty.

$$provision_t = provision_{t-1} + food - 1 \tag{2}$$

This environmental model is the result of our previous investigations with different models and parameters. It ensures that all the tasks will eventually be solved (*requirement 1*)[4]. This may take a long time. However, if there is something in common between the tasks, our system will take advantage of it by reusing it, therefore speeding up the search process by exploiting correlations between tasks.

[3] For these early experiments, we have chosen a very simple reward mechanism. More complicated models will be investigated in our future work.

[4] In some of the previous models, cells tend to specialise in easy tasks only.

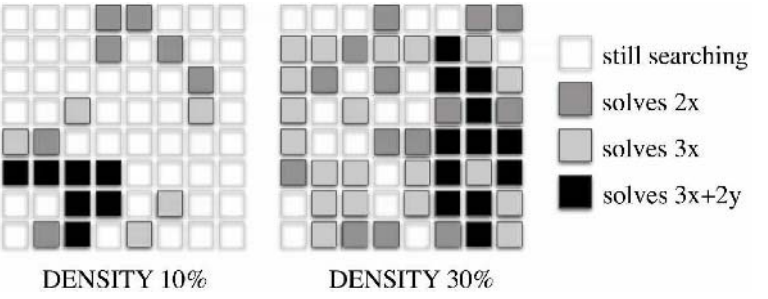


Fig. 6. Example with two different density settings: 10% and 30%. The parameter *DENSITY* enables to hand-tune the maximum percentage of cells that will be able to solve a task.

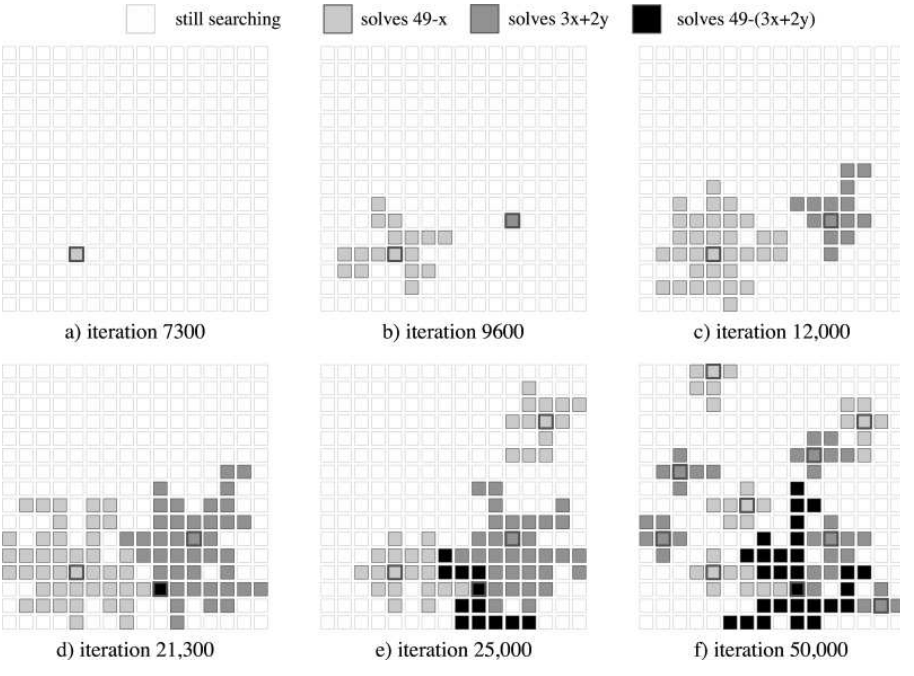


Fig. 7. Knowledge diffusion. Thanks to its parasites, the cell that manages to find the solution to $49 - x$ can diffuse it (a and b). A cell that computes $3x + 2y$ does the same (c). Eventually, in special spots surrounded by solutions to both problems, solutions to $49 - (3x + 2y)$ are likely to appear (d). In the long run, new solutions to $3x + 2y$ and $49 - x$ will appear (e and f). It thus becomes more difficult for parasites to survive. Parasites connected to the solution of $49 - (3x + 2y)$, however, receive more rewards from that solution and won't disappear. Cells with thick border are hosts.

5 Observations

Density. The two main environmental parameters: the resource's quantity and the food needed for a cell to survive can be represented as one parameter $DENSITY$.

$$DENSITY = \frac{QUANTITY}{SIZE}, \tag{3}$$

where $SIZE$ is the total number of cells. This simplifies the model, because only the respective ratio is really important. $DENSITY$ controls the utilisation of the cells in the web. Figure 6 depicts two different settings for that parameter.

This parameter may be tricky to tune, since it involves a trade-off between requirements 5 and 3. Indeed, if it is too small, knowledge won't be accessible (few cells solving tasks). On the other hand, if the density is too large, there will be no room left to solve complex tasks.

Parasites and knowledge diffusion. There is another interesting behaviour of interacting cells that can be observed (see Figure 7). When a cell C_s solves a difficult task for the first time, the solution is almost immediately parasited by its neighbours (Figure 7b). That phenomenon enables to diffuse a solution around the successful cell C_s, thus rendering this solution accessible to an increasing number of cells. Since some cells may need this solution to compute a more difficult problem, knowledge diffusion is highly desirable. Competition between parasites is very intense. They usually appear, survive a couple of iterations, disappear, and after a while appear again, and so on. The dynamism exhibited looks like C_s is trying to reach something in its neighbourhood. For instance, if the diffusion manages to reach the neighbourhood of a cell C_1 that needs it, it will be used and thus the whole chain of parasites from C_s to C_1 will receive a lot of rewards and survive (Figure 7e).

Once some other cells in the web solve the same task as C_s by their own (without parasiting), it becomes more and more difficult for the parasites of C_s to survive (as they always have to share their food with the cells they use). As a consequence, knowledge diffusion will progressively decrease.

Equilibrum/stability. Another parameter, $PROVISION_{MAX}$, has been added. It sets a maximal bound for provisions stored by a cell. Its value highly affects the dynamism of the web. If $PROVISION_{MAX}$ is high, most of the cells are stable and only a few appear and disappear (regime A). If $PROVISION_{MAX}$ is low, we observe much more dynamic structural patterns on the web, with cyclic episodes similar to a kind of *catastrophe* scenario [15]. Good solutions spontaneously appear in the web, and after awhile there are too many cells competing for the same resource. As a consequence, the quantity of the resource they are consuming decreases below 1. Since they don't have enough provisions, they will soon almost all disappear. New cells can then start a new cycle (regime B).

There seems to be no smooth transition between these two dramatically different regimes. Regime A represents a stable and fixed solid state, similar to Wolfram's class

1 of cellular automata (CA) classification [19]. Regime B represents a cyclic state, and is similar to Wolfram's CA class 2.[5]

Robustness. Our system exhibits high levels of robustness. First, multiple solutions to the same task are available in the web. Due to this redundancy, losing one of them is not dramatic. Also, more food will be available for this resource (since $consumers$ have decreased in Equation 1), creating a new solution to replace the lost one (a kind of self-repair). Second, if part of the solution is lost (e.g. one cell dies in a 5-cells solution), the $provision$ variable enables the rest of the solution to survive for a short period, allowing the defective cell to have enough time to recover.

Locality. In living systems, locality has been shown to be an important factor for evolutionary processes (e.g. ecological niches). It also plays a prime role in our system. To compute solutions of higher complexity that reuse previously acquired knowledge, a cell has to be surrounded by useful neighbours (requirement 5). Parameters like $DENSITY$, but also the topology (neighbourhood) or the program length for a single cell, have a direct impact on it. It may be tricky, even impossible, to find a good value for the general case. Issues in scalability are to be expected, too.

The risk is that solutions to simple problems that have to be assembled remain too far from each other. In such circumstances, spontaneous catastrophe events, like regime B, may be useful to reset part of the system when it seems to be stuck in a suboptimal situation.

6 Conclusion

In this paper, we have presented a general self-adaptable architecture designed for multitask learning. The architecture can be implemented in various ways. Our prototype implementation uses the Evolvable Virtual Machines (EVM) [10] framework as the underlying virtual machine. The EVM is itself implemented in Java programming language. The implementation is composed of a network (a web) of interconnected computing programs, cells. Unlike existing cellular models, our cells are capable of universal computation on Turing-like virtual machines, and can modify and manipulate their own code via self-reflection. The cells can also autonomously self-specialise into simpler and more efficient instruction sets, allowing better exploration of the overall search space. Cells contain programs that can call each other, therefore the architecture facilitates collaboration in a symbiotic fashion. This system is self-adaptable in the sense that it can adapt to a dynamic environment without human input and/or predefined behaviour. It is also self-organising, as its internal structure emerges from the symbiotic interactions between cells. These symbiotic interactions enable reuse of previously acquired knowledge. Our architecture is designed to performs well in a multitask context. Up to now, interesting features have been observed at both the cellular and macroscopical levels. This includes, but is not limited to, parasitism, knowledge diffusion,

[5] Wolfram's Classes 3 and 4 can be achieved by tuning an extra parameter $PROVISION_{INITIAL}$, which specifies the initial amount of food a cell receives when it solves a task for the first time.

self-assembly of inter-dependent symbiotic structures, and self-organising utilisation of resources. An interesting phenomenon regarding the cells connectivity has also been observed – the cells connectivity becomes an intrinsic (emergent) resource in its own right. Future research includes investigations of different topologies within cells' local environments. We plan to make our system fully parallel and asynchronous, thus facilitating potential hardware implementations. More extensive and complex experiments with the existing architecture are also planned for the near future.

References

[1] Alberts, Dennis Bray, Julian Lewis, Martin Raff, Keith Roberts, and James D. Watson. *Molecular biology of the cell*. Garland Publishing, 3^{rd} edition, 1994.

[2] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[3] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 2000.

[4] Rich Caruana. Multitask learning. In *Machine Learning*, volume 28, pages 41–75. Kluwer Academic Publishers, 1997.

[5] Radhey S. Gupta and G. Brian Golding. The origin of the eukaryotic cell. *Trends in Biochemical Sciences*, 21(10):166–171, Oct 1996.

[6] Lynn Margulis. *Origin of Eukaryotic Cells*. University Press, New Haven, 1970.

[7] Lynn Margulis. *Symbiosis in Cell Evolution*. Freeman & Co., San Francisco, 1981.

[8] Lynn Margulis and Dorion Sagan. *Microcosmos: Four Billion Years of Evolution from Our Microbial Ancestors*. Summit Books, New York, 1986.

[9] Konstantin Sergeivich Mereschkowsky. Über Natur und Ursprung der Chromatophoren im Pflanzenreiche. *Biol. Zentralbl.*, 25:593–604, 1905.

[10] Mariusz Nowostawski, Martin Purvis, and Stephen Cranefield. An architecture for self-organising Evolvable Virtual Machines. In S.Brueckner, G.Di Marzo Serugendo A.Karageorgos, and R.Nagpal, editors, *Engineering Self Organising Sytems: Methodologies and Applications*, number 3464 in LNAI. Springer Verlag, 2004.

[11] Juergen Schmidhuber. Environment-independent reinforcement acceleration. Technical Note IDSIA-59-95, IDSIA, Lugano, 1995.

[12] Juergen Schmidhuber. A general method for incremental self-improvement and multiagent learning. In X. Yao, editor, *Evolutionary Computation: Theory and Applications*, chapter 3, pages 81–123. Scientific Publishers Co., Singapore, 1999.

[13] Moshe Sipper. The emergence of cellular computing. *IEEE Computer*, 32(7):18–26, 1999.

[14] Christof Teuscher. Information processing in cells and tissues. *BioSystems*, 76:3–5, 2004.

[15] René Thom. *Structural stability and morphogenesis*. Benjamin Addison Wesley, New York, 1975.

[16] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In D. Touretzky and M Mozer, editors, *Advances in Neural Information Processing Systems (NIPS) 8*, pages 640–646, Cambridge, MA, 1996. MIT Press.

[17] Michael D. Vose. *The Simple Genetic Algorithm: Foundations and Theory*. A Bradford Book, MIT Press, Cambridge, Massachusetts/London, England, 1999.

[18] Ivan Wallin. *Symbionticism and the Origin of Species*. Williams&Wilkins, Baltimore, 1927.

[19] Stephen Wolfram. Universality and complexity in cellular automata. *Physica D*, 10:1–35, 1984.

Self-repair Ability of a Toroidal and Non-toroidal Cellular Developmental Model

Can Öztürkeri and Mathieu S. Capcarrere

Natural Computation Group, University of Kent,
CT2 7NF Canterbury, UK
M.Capcarrere@kent.ac.uk,
http://www.cs.kent.ac.uk/research/groups/aii/ncg/index.html

Abstract. This paper is part of a larger project whose main objective is to demonstrate experimentally that the following hypothesis holds: *computational developmental systems on a cellular structure are a) naturally fault-tolerant and b) evolvable*. By naturally we mean that the system is not fault-tolerant by explicit design nor due to evolutionary pressure, but rather that the framework insures a high probability of fault-tolerance as an emergent property. In this paper, we propose to study the self-repair capacities of a specific developmental cellular system introduced in [13]. More specifically we compare the toroidal and the non-toroidal cases. Their evolvability is to be presented in details in a further article. All the examples studied here have been evolved to configure an abstract digital circuit. The evolved organisms are subjected to a series of different fault models and their self-repair abilities are reported. From the results exposed here, it can be concluded that, while not systematic, perfect self-repair, and hence fault-tolerance is a highly probable property of these organisms and that many of them even exhibit fully perfect self-repair behaviour under all tests performed.

1 Introduction

Obtaining resilient computing systems that self-organise around faulty elements to keep on functioning has great potential benefits but remains hard to design by hand. Besides, hand-designed systems, while usually being provably safe on foreseen errors, are often brittle when encountering unplanned errors. Biological systems, on the other hand, display a great adaptability to changing conditions and, in general, great resilience to harsh and uncertain environments. The evolutionary reasons for this state of facts are obvious and one approach to obtain resilient computing systems, if one is ready to forego the provability element, is to evolve programs in uncertain/changing/faulty environments. Their resilience is then the by-product of the evolutionary process. While this has had some successes in the past [15], constraints on time and resources (and maybe more fundamental reasons) have limited the results obtained using this strategy to a few very specific examples. Another means of adaptation in biological systems is development. Continuing the bio-inspired route, we propose in this paper to experiment a framework based on development and cellularity to obtain resilient computing systems.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 138–148, 2005.

In this paper, we present encouraging recent investigations on a developmental structured cellular system in terms of evolvability, that is:– *abstract*, i.e., we do not pursue biological realism; – *usable*, in the sense that it is evolvable; – *developmental*, in the sense that it is able to self-configure from minimal or no environmental information; – *stable*, in the sense that it is actually continuously configuring itself, the working configuration being a stable state of development; – *cellular*, it is based on a cellular structure where each unit is relatively simple and all units are identical, though they may differentiate; – *structured*, i.e., the neighbourhood of the units is fixed.

We will first quickly review existing systems in section 2 and present the system and the experimental set-up in the following section. In section 4, we then present the experimental results averaged on almost 100 toroidal and 100 non-toroidal organisms successfully evolved to configure a circuit that fits a given random truth table. From these, we establish the main strength and weaknesses of such systems, and delineate rough probabilities to obtain fully "perfect" self-repair ability within such a framework. While concentrating on a specific model of developmental system, the breadth of experiments presented here allows one to conclude that they go beyond the anecdotal or the circumstantial.

2 Background

Growth is usually defined in biology as "a purely biological unfolding of events involved in an organism changing gradually from a simple to a more complex level". Within the realms of this paper we will consider growth to be the unfolding of events that leads a system, more specifically a cellular system, to configure its global state in a *complex*, *stable* and *useful* pattern. *Complex* means that the system should start with a minimal amount of environmental information compared to the information in the 'adult' state. *Stable* means that the growth process should stop by itself, or rather, that the process continues, but does not alter the global state of the system anymore. This is very much like development in biology: once an organism has reached its adult size, cells are renewed continually, but the organism as a whole remains (more or less) unchanged. Finally, *useful* means that this final configuration of the whole system fits some prerequisite needs, which may be a pattern to display, but also the configuration of an FPGA, or even that given some input and output cells, the system is able to compute some useful functions. These three characteristics are at the heart of the system. We can note here that while biology is an inspiration, we are not aiming to model it. In fact, Lewis Wolpert [20] has explicitly argued against the idea that reconstruction is a growth process. However these considerations do not affect *our* interpretation of growth. Our main aim in the long run is to obtain a practical, usable software framework. Growth in its most general meaning has recently attracted a number of studies, ranging from biological modelling [2, 18] to abstract developmental systems [1, 8, 14] via morphogenesis [4, 10, 17]. The reasons for this interest are multiple, ranging from code compactness and scaling problems[17, 19] to the biological understanding of development[3, 5]. As

we will detail below, some studies have tried to exploit growth as a means of self-repair [6–8, 16].

Fault-tolerance & self-repair: A fault-tolerant system is a system that functions as specified despite the presence of faults. This is often achieved by some sort of redundancy in the system, thereby allowing the demise of faulty parts. This redundancy may be explicit such as twin-engine aircraft able to fly on one engine, or implicit, such as the evolved messy gates circuit [11] where the circuit is not explicitly evolved to be fault-tolerant, but experience proves that parts may be removed without affecting the function of the circuit. A *Self-repairing* system is a system that is able to "reconstruct" a faulty part so that function is restored when a fault occurs. Hence, self-repairing systems are by definition fault-tolerant. Existing self-repairing systems in computer science obviously do not physically reconstruct, but rather reconfigure available spare parts to reconstruct the original circuit/logic/etc. This project concentrates on reconfiguration. Fault-detection is not a prerequisite of fault-tolerance, but is a necessity to trigger a self-repair. This detection is either done by knowledge of correct outputs or manual intervention. In contrast, the framework presented here removes this major weakness through a continuous growth process.

Existing developmental cellular models for self-repair: A few abstract, non realistic, developmental models on a cellular substrate have been researched with the aim of gaining self-reconstruction or self-reorganisation to handle faults. We propose in this subsection to briefly review them and highlight the differences with our system. In [9], Miller and Thompson present a further study of the system developed earlier [10] which is a rather complex abstraction of the growth process based on the diffusion of chemical within a CA-like structure. This research has obtained interesting results in terms of morphogenesis and self-repair, however it lacks greatly in both stability and the quality of the self-repair. Our current results tend to show that simpler systems actually perform better. Nevertheless it is the first attempt that we know of that tries to assess, albeit in a limited way, its evolvability. Liu *et al* [6] try to evolve a morphogenetic system but on a hardware base (FPGA), an interesting cellular base. The model is similar to that of Miller but goes further in terms of functionality. A single example shows that the system presents some self-repair capability. This result illustrates the potentials but is a long way away from demonstrating the property, being based only on one experiment on one organism. A very interesting approach is that of Macia and Durbeck [7] where they present the design of an FPGA that is able to completely (re)configure itself on spare parts. Nevertheless it obtains its perfect error recovery at the price of requiring a full specification of its expected input/output pattern to detect that an error occurred, thereby implying a great waste of the logic available. Another interesting hardware system is that of Mange *et al*, [8], but it also implies explicit fault detection. Finally, to complete this rapid tour of the few works within the field, Roggen and Federici [16] compared different types of developmental process in terms of scalability and robustness, but their study is also very limited.

3 Experimental Set-Up

The cellular and developmental model: The model is an extension of the well-known model of cellular automata. The system is both discrete in space and time. Each cell lies at the vertex of a 2-dimensional lattice and possesses a x-bit state that is readable *and modifiable* by any one of the 8 neighbouring cells (Moore neighbourhood). x is dependent on the given problem. All cells act according to the same *memoryless* program whose inputs are the $9x$ bits of the states of the cell itself and of its 8 neighbours. Like a classical CA, the result of this internal program execution determines the new state of the cell. However, unlike the CA model, this cell may also alter the state of its 8 neighbouring cells. Given that all the states' bits are considered individually, this internal program could be best described as a $9x$-bit input, $9x$-bit output function.

Both toroidal and non-toroidal grids are used. In the non-toroidal case, to the program, the "virtual" cells beyond the borders appear to have a 0^x state that cannot be altered by a neighbour. This 0^x state is not specific to border cells, and is thus indistinctive.

The neighbour's state rewriting property, which is at heart of the whole process, entails a series of questions on the order of update of the cells. It forbids a fully parallel update. A single model of asynchronous updating was studied up to now. Each cell reads in parallel its neighbourhood and tries to write its state and its neighbours' state in parallel. However, to solve the problem caused by many cells wanting to rewrite a cell's state, only the right-most, bottom-most cell requiring an update has priority. This is a fully deterministic model. A more interesting model from a research viewpoint, but less practical from a hardware viewpoint, would be a random asynchronous update mode. This is currently under study.

The developmental aspects rely on that rewriting property, but more fundamentally in the way the system is both evolved and tested. In effect the state of all the cells in this grid start at 0^x, where x is the length of the state of each cell. The disruption in the uniformity, necessary to obtain non uniform configuration of the grid, is due to this rewriting priority which implicitly defines a relative order. This disruption in uniformity is sufficient to obtain any kind of configuration. This is helped in the non-toroidal case by the borders which also create a non-uniformity. The fact that the circuit configures itself from minimal environmental information is essential to obtain resilient circuit. Development in this context is thus really self-organisation.

The evolutionary algorithm: All solutions studied in this paper were evolved using a straightforward extension to Cartesian Genetic Programming (CGP), coincidentally first designed to evolve digital circuits [12] and thus well suited for problems involving binary inputs/outputs. It presents the great advantage of not suffering from the bloat problem.

Each individual represents the program to be executed in all the cells of the system. Basically we have to evolve a $9x$-bit input, $9x$-bit output program. Each bit is considered as a separate input/output. We refer the reader to [13] for the precise setting, the evolvability not being our object in this paper.

Tasks & Faults Models
The systems subjected to faults have been evolved to configure an abstract 3-input, 2-output, circuit model to fit a truth table. Ten random truth tables have been chosen as goals. In this task, the fitness is judged on the functionality of the circuit, not on its specific configuration, while its repair ability is judged on its specific configuration not only the functionality of the circuit. This circuit is tested on both toroidal and non-toroidal grids. We first describe the circuit model and then detail the fault-model to which we subjected the organisms evolved.

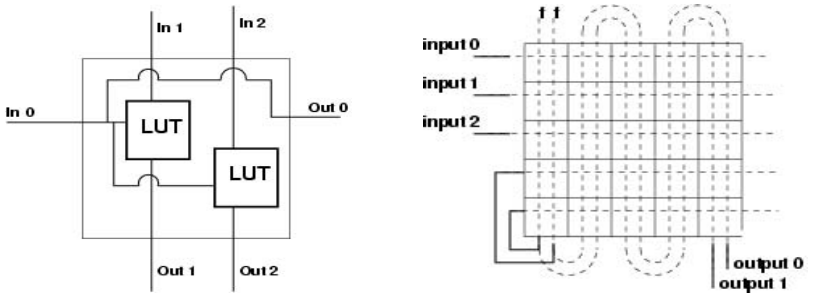
The digital circuit abstract model: The developmental system presented earlier rely on a cellular structure. It is natural to design a circuit model topologically equivalent. The circuit model therefore relies on basic units (see fig.1.a) arranged along a cellular structure (see fig.1.b). Straightforwardly, each basic unit is then configured by the state of its corresponding cell in the developmental model. Conceptually, it is important to distinguish the two-layers. One is the developmental model, which, when run for a certain number of time steps, reaches a stable state. In this stable global state each cell has its own x-bit long state. The second layer is the circuit. Each basic unit of the circuit behaviour is determined by an x-bit long configuration string. As the first and second layer are homoeomorphic, the state of the cell of the first layer is the configuration string of the corresponding basic unit of the second layer. Hence, during the developmental process, the circuit is not yet configured in a working state. Obviously, these two layers can be merged. In some experiments, a further few bits are used in the developmental layer. These are free bits that can be used for development but do not affect the configuration of the circuit.

The basic unit (fig.1.a) used for the results presented in this paper has n vertical lines and 1 lateral line coming in. Whatever its configuration, each unit propagates the lateral signal through, unchanged. The number of vertical inputs, n, is the same as the number of global outputs of the whole circuit. While this restricts the complexity of the possible circuits, it already allows for reasonably complex functions, as a 16-bit or 32-bit bus would not be unrealistic.

The configuration string of each unit is $4 + n$-bit long. For each incoming wire, a bit determines whether a look-up table (LUT) is applied to it or whether the signal is just propagated through. If a LUT is applied, the inputs to this LUT are the lateral input and the vertical line concerned. 4 more bits determine the outputs of the LUT. The same LUT is applied to all the wires.

As can be seen in figure 1.b, the vertical lines run through the whole circuit and its values out of the last unit are read as the output of the whole circuit. Each basic unit is evaluated sequentially from top-to-bottom and left-to-right. For the experiments described in this paper, the value of the two vertical lines at the end of the first column are fed back into the circuit as lateral inputs.

The fault model: In any studies wishing to assess robustness and self-repair qualities, a precise fault-model should be defined to delineate the scope of validity of the study. The fault model used in this paper is at the same time simple and rather exhaustive. First, we only test here self-repair abilities concerning the configuration process. In fact, it could be said that we test the self-



(a) (b)

Fig. 1. In (a) one can see the model of a basic element of the circuit. In this example, each of them have 3 inputs and 3 outputs. It always propagates its lateral input (In 0). Its configuration string determines what the look-up table is and whether or not it is applied to each of the wire. In (b), one can see a full 5x5 circuit, with 3 global inputs and 2 global outputs. **f** denotes a constant false value. The circuit is evaluated sequentially top-to-bottom, left-to-right.

reconfigurability process. Therefore we are not concerned with faults in the mapping configuration-function and assume this mapping is perfect, which is by far not unreasonable. Within the scope of our study, the main limitation of our fault model is that it is a strike-once model. After a fault occurred, the reconfiguration of the circuit, the repair phase, is done in a safe environment. Again such a model is perfectly reasonable as long as this reconstruction process is fast and *perfect*.

The fault-model simply consists in choosing a certain number of cells affected by faults. A cell can be affected by 2 types of faults: – *reset disruption*, the state is reset to 0^x; – *random disruption*, the state of the cell is set to any random value. These models may seem over-simplified, however they encompass many faults. First, obviously a fault in the workings of the internal cell program that would entail writing the wrong state for itself and/or for its neighbours. Second, it also models a reading error. In effect if a cell misreads its environment, it takes a wrong decision (in interesting cases), and therefore the circuit has at the next time step one or more cells in the wrong state. Finally it also models writing errors. The number of cells affected are 1, 4 in a 2x2 block, and 5 randomly chosen cells. In this paper, each successfully evolved organism was subjected to 4 kinds of faults: a one-cell reset, a five cell reset, a five cell random fault and a 2x2 block random fault. Therefore while gentle time-wise, this error model is rather complete space-wise and working-wise.

4 Results

While previous studies have often concentrated on one or a very few numbers of individuals, the results presented here have been established on roughly 100 individuals for each of the set of experiments to give some generality to the results found. In the same way, rather than choosing a specific task, to avoid

bias, all experiments are averaged out over 10 different random version of the goal. The aim of the evolutionary runs is to find an organism that develops into the configuration of a circuit (see fig. 1) whose behaviour fits a given truth-table. This was repeated 10 times over 10 randomly chosen truth tables. These 100 experiments were repeated twice, once with a non-toroidal developmental grid, and once with a toroidal one.

Random truth tables experiments: The task here is to configure the circuit presented above so that its behaviour fits a given complete 3-input, 2-output truth table. The circuits are made up of 5x5 basic units. Each basic unit of the circuit needs 5 bits to be configured. Hence a 120 bits configuration is needed to configure the whole circuit. The developmental layer is thus 5x5, but each cell contains 7 bits. The first five bits configure the circuit unit while the remaining 2 are ignored by the circuit layer and can be used freely for the developmental process. The fitness for the evolutionary runs is how well the circuit, configured by the adult organism, i.e, once it has reached a stable state, fits the truth table. For this task two models of configuration grids are studied, a toroidal and a non-toroidal one. Ten evolutionary runs were done for each of the truth table.

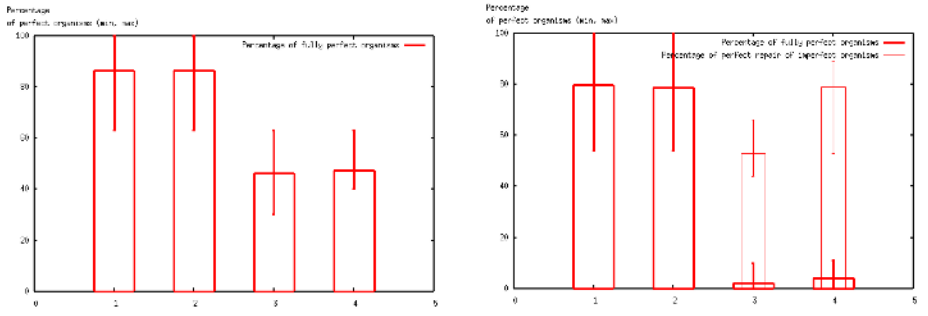
Each grid is let to develop for 7 time steps before the fault strikes. There is no specific fault signal, and it is only the perturbations provoked by the fault that trigger the healing process.

4.1 Non-toroidal Grids

Firstly it can be noted that whatever the truth table, the system is highly evolvable as 99 out of the 10x10 runs ended up with a successful organism, thereby giving us a reasonable sample to test out the self-repair ability of the model.

Each of the 99 organisms were subjected to the 4 kinds of faults described in subsection 3, 1000 times, except for the reset-one-cell fault which is tested exhaustively on the 25 possibilities. Even though only 120 bits out of the 175 bits are used for the circuit layer configuration, the repair process is considered perfect if and only if the whole 175 bits are restored to their original adult/stable state. Hence it is often the case that even when perfect self-repair is not achieved the circuits keep on functioning as normal, but this is not our main concern in this paper. Perfect self-repair provides several advantages, including, no degradation of the performance in the long run and generalisation of the results to tasks where all the bits are necessary.

As one can see in figure 2, the amplitude of variation between truth tables is minimal. This entails that the self-repair ability observed results from the nature of the system rather than the specific task and results can be expected to replicate for any other truth tables. The second striking fact is the really extraordinary capacity of these organisms to cope with the reset faults. If one only considers organisms that repair themselves perfectly in all of the 25 one-cell reset cases (fig. 2(a), col 1) and all of the 1000 5-cell reset cases tested (fig. 2(a), col 2), we still have 88% of the organisms evolved, and this result does not vary much from one truth table to another, as the lowest one is 63% and



(a) (b)

Fig. 2. For both graphs, in bold are the percentage of fully perfect organisms that self-repair perfectly in all test cases; in column 1, for the 1-cell reset test (x25), in column 2, for the 5-cell reset (x1000), in column 3, for the 5-cell random disruption (x1000), and in column 4, for the 2x2-cell block random disruption (x1000). The error bars highlight the minimum and maximum value found. In figure (a), results for the non-toroidal grid are represented. In figure (b) results for the toroidal grid are presented. In that second figure, for exp 3 and 4, the thin line plots are the average number of times the organisms self-repair perfectly out of the 1000 tests.

highest is 100%. The result is particularly surprising as none of the organisms were specifically evolved to display that self-repair property. Nevertheless as the organisms develop from an empty grid these resetting faults are gentler than purely random disruption. This can be checked in the lower percentage of fully perfect organisms found for random errors (fig. 2(a), col 3 & 4). This said, even in these harsher conditions, the resilience abilities are still striking as 46.3% of the organisms fully repair in all the 1000 5-cell random disruption tests (fig. 2(a), col 3) and 47.3% of the organisms fully repair in all the 1000 2x2-block random disruption tests (fig. 2(a), col 4). As could be expected, all organisms perfect in these latter experiments behave perfectly in the former experiments. Hence almost half of the organisms evolved for circuit configuration display absolutely perfect self-repair behaviour on all the 3025 tests, every time restabilising into their stable, original, working conditions. It is interesting to note that if we look at the organisms that fully repair themselves but not in all of the 3025 tests, there is still a high degree of resilience exhibited. For instance, the average percentage of perfect self-repair out of 1000 5-cell random disruption tests among *non-fully* perfect organisms is still 56.1%.

4.2 Toroidal Grids

Toroidal developmental grids do not display much difference in terms of evolvability as 98 organisms out of the 10x10 runs were successfully evolved. This success allows us to test their self-repair abilities on a significant sample. To allow meaningful comparison with the non-toroidal case, the 10 truth tables chosen at random above were reused for this experiment.

As one can see in Figure 2.(b), in columns 1 and 2, the ability to cope with reset type errors is also very high in the toroidal case. 78.7% of the organisms evolved self-repair perfectly in all of the 1000, 5-cell reset tests. As before, this means that one using this framework is very likely to evolve not only good, but fully perfect organisms for this type of error. Interestingly the overall average in the toroidal-case is about 8% less than the average in the non-toroidal case, and the minimal value found for one truth table for the former is 9% less (53%) than the minimal value found for one truth table for the latter. This tends to confirm that, while good, the toroidal case is not as good as the non-toroidal case to produce resilience. Another interesting fact emerges from this comparison of minimal value: they are not happening for the same truth table. This further reinforces the idea that the fault-tolerance results exhibited here are totally independent on the function of the circuit sought.

The excellent ability to cope with errors however collapses when one considers the more general random disruption. In that case only 2% (2 organisms) exhibit fully perfect self-repair for the 1000 5-cell random disruption test case, and a slightly better 4% for the 2x2 block random disruption case. These results bear no comparison with the non-toroidal ones. The reasons for this difference are hard to assert with any degree of certitude. Obviously the first idea that springs to mind is that borders favour stability, thereby favouring self-repair. However, the results in terms of evolvability contradict somehow this hypothesis. In effect, evolution is considered successful if the organism develop into the required configuration of the circuit, but only if it does so in a stable manner. Here, unlike in other works, there is no explicit growth phase with a stopping time. The process stops itself by reaching a configuration that is stable. If toroidal grids were truly more unstable than their non-toroidal counterpart, their evolution should prove harder[1], which is not the case both in terms of the percentage of successful runs and in terms of the speed of the evolution. Besides, becoming stable too fast, i.e, stable in a wrong state, is not better. If the reasons of these differences remain unclear at the moment, we should moderate them. The fact that we compare only fully perfect organisms discard almost perfect ones, such as an exemplar that self-repair perfectly in 999 of the 1000 test cases for the 5-cell random disruption test. Actually if we look at the average percentage of perfect self-repair out of the 1000 tests (figure 2.(b), col. 3 & 4, thin lines), the inability to cope with random disruption is mitigated. On average, the organisms self-repair perfectly half the time (52.8%) for the 5-cell case, and three-quarter of the time (78.9%) of the time for the 2x2 block case. More importantly, the worst result for a truth-table is 44.1%, and for a single individual 21% while for the best table it is 66.3% and 100% for the best individual. Finally if one looks at the block random disruption, this spread of the self-repair ability is even clearer, with at worst for one truth table, perfect self-repair in 42% of the tests for the worst organism. This highlights both the very good ability to cope with strong disruption even in the toroidal case, and that this ability is well spread among all the organisms found.

[1] one should note that if the task evolved is too simple, this weakness may not appear.

5 Concluding Remarks

While these results are preliminary given the sample, their consistency across all organisms successfully evolved allows us to draw some conclusions.

Firstly, developmental cellular systems always exhibit some self-repair ability. Within a few evolutionary runs, it is possible to find, with a very high probability in the non-toroidal case, a fully perfect organism that is stable even under the worst conditions. This obviously calls for further investigations on bigger grids and for different tasks. Secondly, it is clear that a non-toroidal grid should be adopted within a framework that has resilience as an aim. While results for toroidal grids are not bad in themselves, they do not bear comparison with the non-toroidal case. Nevertheless, this advantage on resilience may be lost on the evolvability ground with bigger grids, where signal flow is more important.

There are still many questions that remain open. Most notably the decoupling between the developmental layer, and the circuit configuration, in other terms between the developmental process and the fitness criteria may be one of the important factors both in terms of evolvability and of self-repairability. The number of stable patterns that give rise to the correct output is surely enormous. Therefore, one may hypothesise that it leaves a lot of room for the evolution to fall into a large basin of attraction that configure the circuit properly. This facilitates the evolution but also surely the re-stabilisation after perturbation. A pattern as a goal would introduce a lot more constraints in the evolutionary runs, and may be harder to evolve, and the evolved organisms may be brittler. Results do not allow us yet to answer this question.

References

1. D. Basanta, M. Miodownik, P. J. Bentley, and E. Holm. Evolving and growing microstructures of using biologically inspired ca. In *2004 NASA/DoD Conference on Evolvable Hardware*, pages 275–282, Los Alamitos, CA, 2004. IEEE Comput. Soc.
2. P. Hogeweg. Shapes in the shadow: evolutionary dynamics of morphogenesis. *Artificial Life*, 6(1):85–101, 2000.
3. J. Kim. transsys: a generic formalism for modelling regulatory networks in morphogenesis. In J. Kelemen and P. Sosik, editors, *Advances in Artificial Life. 6th European Conference, ECAL 2001. Proceedings*, volume 2159 of *Lecture Notes in Artificial Intelligence*, pages 242–251. Springer-Verlag, 2001.
4. O. Kniemeyer, G. Buck-Sorlin, and W. Kurth. A graph grammar approach to artificial life. *Artificial Life*, 10(4):413–431, 2004.
5. C. Leung and M. Berzins. A computational model for organism growth based on surface mesh generation. *Journal of Computational Physics*, 188(1):75–99, june 2003.
6. H. Liu, J. Miller, and A. Tyrell. An intrinsic robust transient fault-tolerant developmental system. In *Workshop on Regenerationand Learning in Developmental Systems, GECCO'04*, 2004.
7. N. J. Macias and L. K. Durbeck. Self-assembling circuits with autonomous fault handling. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. Salem Zebulum, editors, *Proceedings of the 2002 NASA/DoD Conference on Evolvable Hardware*, pages 46–55, 2002.

8. D. Mange, M. Sipper, A. Stauffer, and G. Tempesti. Towards robust integrated circuits: The embryonics approach. *Proceedings of the IEEE*, 88(4):516–541, April 2000.
9. J. Miller and P. Thompson. Beyond the complexity ceiling, evolution, emergence and regeneration. In *Workshop on Regenerationand Learning in Developmental Systems, GECCO'04*, 2004.
10. J. F. Miller. Evolving developmental programs for adaptation, morphogenesis, and self-repair. In W. Banzhaf et al, editor, *European Conference on Artificial Life VIII (ECAL'03), the proceedings of*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 256–265, Berlin, Heidelberg, 2003. Springer-Verlag.
11. J. F. Miller and M. Hartmann. Untidy evolution: evolving messy gates for fault-tolerance. In Y. Liu, K. Tanaka, M. Iwata, and T. Higuchi, editors, *Proceedings of the 4th Conference on Evolvable Systems (ICES'01)*, volume 2210 of *LNCS*, pages 14–25. Springer-Verlag, 2002.
12. J. F. Miller and P. Thompson. Cartesian genetic programming. In *European Conference on Genetic Programming (EuroGP'00), the proceedings of*, volume 1802 of *Lecture Notes in Computer Science*, pages 121–132, Berlin, Heidelberg, 2000. Springer-Verlag.
13. C. Ozturkeri and M. S. Capcarrere. Emergent robustness and self-repair through developmental cellular systems. In *Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*, pages 21–26. MIT Press, 2004.
14. R. Pfeifer. Interacting with the real world: design principles for intelligent systems. In *Ninth International Symposium on Artificial Life and Robotics (AROB 9th'04)*, volume 1, pages 13–18. Oita University, 2004.
15. L. Righetti, S. Shokur, and M. S. Capcarrere. Evolution of fault-tolerant self-replicating structures. In W. B. et al, editor, *European Conference on Artificial Life VIII (ECAL'03), the proceedings of*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 278–288, Berlin, Heidelberg, 2003. Springer-Verlag.
16. D. Roggen and D. Federici. Multi-cellular development: Is there scalability and robustness to gain? In E. Yao, X.and Burke, J. Lozano, J. Smith, J. Merelo-Guervs, J. Bullinaria, J. Rowe, P. Tino, A. Kabn, and H.-P. E. Schwefel, editors, *Proceedings of PPSN VIII*, volume 3242 of *LNCS*. Springer-Verlag, 2004.
17. K. Stanley and R. Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003.
18. F. Streichert, C. Spieth, H. Ulmer, and A. Zell. Evolving the ability of limited growth and self-repair for artificial embryos. In W. Banzhaf et al, editor, *European Conference on Artificial Life VIII (ECAL'03), the proceedings of*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 289–298, Berlin, Heidelberg, 2003. Springer-Verlag.
19. P. van Remortel, B. Manderick, and T. Lenaerts. Gene interaction and modularisation in a model for gene-regulated development. In *2004 NASA/DoD Conference on Evolvable Hardware*, pages 253–260, Los Alamitos, CA, 2004. IEEE Comput. Soc.
20. L. Wolpert, R. Beddington, T. Jessell, P. Lawrence, E. Meyerowitz, and J. Smith. *Principles of Development*. Oxford University Press, 2nd edition, 2002.

Simulating Evolution with a Computational Model of Embryogeny: Obtaining Robustness from Evolved Individuals

Chris P. Bowers

School of Computer Science, The University of Birmingham,
Birmingham, B15 2TT, UK
C.P.Bowers@cs.bham.ac.uk

Abstract. An evolutionary system is presented which employs an embryogeny model to evolve phenotypes in the form of layout of cells in specific patterns and shapes. It is shown that evolved phenotypes exhibit robustness to damage. How and why these traits appear is discussed and it is conjectured that it is the result of the effects of a complex mapping upon simulated evolution.

1 Introduction

Most simulated evolutionary systems use the idea of survival of the fittest to perform directed searches toward optima. An individual in an evolutionary process has only one aim - to survive and propagate. In order to do this it must be capable of adaptation. The most obvious form of adaptation is through beneficial mutations or exchange of useful genetic material with other individuals. This is dependent on the genotype, the representation used by an individual within the evolutionary process. However, this is not the whole story since there is a second form of adaptation which is heavily used by natural evolution, of which humans could arguably be considered the pinnacle. Phenotypic adaptability defines the capability of an individual to be robust to varying environments and circumstances. This adaptability is expressed through the mapping of the genetic representation of an individual to its physical representation or its phenotype. In essence it is the phenotype which determines the likelihood of an individual propagating further through the evolutionary system and it is the genotype which determines how this propagation is conducted. Therefore, ensuring the survival of the phenotype inherently results in the same for the genotype. This has largely been ignored by the evolutionary computation field since the genotype is often considered as the direct basis for representation of a potential solution.

This mapping is a product of developmental growth processes and as such can be considered complex relative to the type of representations traditionally used for simulated evolution. With the existence of such complex mappings, for evolution to be anything more than a blind search, it must find a way to relate useful phenotypic traits to structures in the genotypic representation. The work conducted here investigates some of the evolutionary effects of introducing such a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 149–158, 2005.

mapping process between genotype and phenotype. In particular, the ability for a phenotype to be adaptable to change, measured here as robustness to damage, is examined.

2 The Embryogeny Model

The mapping from genotype to phenotype in natural evolution is a developmental growth process. This work considers the initial part of this process, embryogeny, which describes the growth of an embryo from a single stem cell to a functioning multi-cellular organism. This form of developmental modelling has been termed computational embryogeny [6] or embryonics [8,11]. As with any such model a number of abstractions and assumptions are required. The critical point is to find a suitable way of representing what can be observed in nature such that it is both computationally feasible and yet still exhibits the required characteristics that make it useful to model.

2.1 The Physics

The model exists in a real valued 2-dimensional Cartesian space. The embryogeny mapping requires that the phenotype consist of discrete processing blocks (cells). Each cell is modelled as a circle with a given radius and can exist at any real valued position. This allows the potential for cells to overlap and so a simple Verlet algorithm [13] is used to shuffle cells in a realistic fashion to minimise overlap. The state of a cell is primarily described by a set of 20 chemical concentrations independently stored for each cell. Cells can only interact either through physical forces modelled by the shuffle algorithm or through the diffusion of these chemical states. Chemical diffusion is modelled as a simple Guassian distribution which ensures that the diffusion of a specific chemical from a specific cell can be calculated instantaneously for any point in the 2-dimensional space. The growth process itself is modelled in parallel across all cells and calculated in n discrete growth steps. A cell can only divide once at each step so there is the potential for a total of 2^n cells. Division is directional with respect to a division spindle vector stored independently in each cell. When a cell divides it produces a daughter cell which is an exact replica in every aspect except position.

However efficient the code may be, computational costs will obviously increase with the number of cells in the model and so the number of iterations of growth, for the purposes of evolution, has been fixed to $n = 7$. Also the environment in which these cells are modelled is of fixed size preventing cells from extending outside of this range and this is observed by the shuffle algorithm. The simplest way to enforce this is to define a ratio between cell radius and boundary size, in this case a ratio of $20 : 1$ allows a string of 11 cells in width or height as measured from the cell centres. In this work this ratio is fixed across all cells ensuring they are each of equal size and shape.

The work presented here specifically investigates various patterns and shapes that cells can form. This requires some method to enable the visualisation of

differences between cells. Three of the chemical concentrations stored within the cell are used to represent red, green and blue values for visualisation purposes. Also two further chemicals are utilised for spatial differentiation as orthogonal morphogens carrying both vertical and horizontal gradients [1]. The remaining 15 chemical concentrations are sufficient to store any variables which may be utilised for pattern formation in whichever way evolution deems useful.

Overall, this approach offers significant improvements over previous problems encountered with such models by removing issues such as cell overwriting, expensive diffusion algorithms and serial artefacts [3,9].

2.2 The Genetics

The Operon model is a tool used by biologists to explain how genes form networks of complex interactions termed Genetic Regulatory Networks (GRN) [1]. In this work a much simplified version of the Operon model is used as a basis for the genetic representation. As shown in figure 1, the genome can be broken down into a set of genes each represented as a pair of integers, the first describing the genes function A and the second the dependent protein B.

The genome is processed as follows. The first gene in the genome is always expressed and so its function A is carried out dependent upon the value stored for the concentration of protein B in that particular cell. A gene's function can be to alter the local cell state and/or to control the expression of the next gene in the genome. In this manner the genes are linked in a chain of expression dependent upon their position in the genome. There are 26 genes in total consisting of the following functions:

- Terminal genes are always expressed. They end a current chain of expression and begin a new chain by controlling the expression of the next gene directly without dependence on the cell state.
- Expressive genes are similar to terminal genes but their function is dependent upon a given protein. This allows the cell state to determine whether or not a chain of expression becomes active.
- Evaluative genes are dependent on expression from a previous gene. They control the expression of the next gene based on some evaluation performed upon a given protein. These genes control the expression within a group of genes and can be chained together to form more complex functions.
- Functional genes consist of those genes that can be controlled by expression but their function is to directly alter the cell state such as to cause the cell to move its division spindle, divide or die.

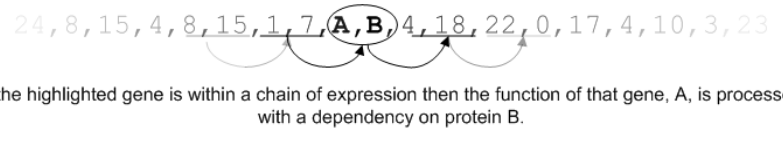


Fig. 1. The genetic representation showing gene structure and chain of gene expression

Each cell in the model has an identical genome so differences in cell behaviour directly relate to differences in cell states. Therefore two cells with identical cell states will perform identical actions.

3 The Evolutionary Algorithm

We can imagine genotype space consisting of all possible ways of constructing a genome from the set of available genes and phenotype space consisting of all possible ways of constructing all possible cell states into some form of pattern. Considering that these two spaces are linked by the embryogeny mapping process there are two points to be considered which may have a direct impact on evolutionary performance.

Mapping Coverage is a potential issue since the phenotype space is likely to be much larger than the genotype space. This problem is further compounded by the opportunity for multiple redundancy in the genetic representation which causes large numbers of many-to-one mappings from genotype to phenotype. The result is that genotype space will only map to certain areas of the phenotype space and leave areas of the phenotype space unmapped and therefore inexpressible by the genetic representation. It is imperative to ensure that global optima in the phenotype space are expressible by this mapping.

Neighbourhood structures at the phenotypic level produce smooth fitness landscapes since it is in this space that an individual is evaluated. However, the mapping process does not conserve neighbourhood structure and so when fitness values are mapped back to genotype space the resulting fitness landscape may be very noisy and discontinuous.

A difficult problem associated with evolving a complex mapping, such as the one described here, is the computational cost of performing the genotype to phenotype mapping. In order to overcome this several computational clusters were utilised with a simple parallel genetic algorithm [10]. This is realised through a distributed fitness evaluation using a simple server-client architecture in which evolution is conducted upon the server. When a population evaluation is required the population is simply divided into a set of sub-populations which are then passed to and then processed by a set of client machines before being returned and combined to recreate the original population. For the work presented here 10 client machines were used with a total population size of 200.

Another difficulty of a complex mapping is that the many-to-one relationship between genotypes and phenotypes leads to the potential for large numbers of genotypes to map to identical phenotypes and thus have equal fitness. The overall result is, that for a representation such as this, the evolutionary search space is fractured into networks of neutrality [5]. If a population occupies one of these neutral networks then selection pressure becomes redundant. It is important therefore to ensure that the population avoids stagnation in these areas. By

using random selection and continued use of search operators a neutral drifting of the population can be achieved.

A simple genetic algorithm is utilised which selects the best 50% of the population. Each of these selected individuals is subjected to a random two point crossover operation with another of the selected individuals. The product of this crossover is then subjected to a further random single point mutation. The resulting offspring then replace the worst 50% of the population. This ensures a consistent generational selection pressure during adaptive evolution whilst also enabling random selection during neutral evolution. Fitness of an individual is evaluated as the number of cells present in a phenotype which are correctly spatially differentiated according to a predetermined target template.

4 Simulation Results

The aim of this work is to investigate how introducing an embryogeny mapping process affects the robustness of an evolved individual. Several templates are utilised against which evolved individuals are evaluated (figure 2). This allows for the analysis of how robustness varies dependent upon the actual structure of the target phenotype.

For each template two results are recorded. The phenotype robustness is a measure of the ability of a phenotype to repair from damage. This is calculated as the percentage of cells which match the original template after first culling cells randomly from a fully developed phenotype until a given percentage of cells are removed, in this case 50%, then reapplying the embryogeny growth process until cells have recovered to their previous numbers. Genotypic robustness measures the effect of single point gene mutations on the overall phenotype. A single gene in the genome is mutated and then a phenotype is grown and the percentage of correct cells is then calculated. For each evolved genome, these values are taken over an average of 100 samples. For each template, 10 genomes were examined. These values are also exhibited in figure 2 along with their standard deviations.

Figure 3 shows the typical behaviour of a phenotype in response to damage corresponding to the template show in figure 2D, at various stages of growth and repair. In this example the phenotype is grown to be much larger and contain more cells than it was originally evolved for, highlighting the scalable capabilities of evolved solutions. The model is also capable of handling overgrowth (figure 3I), where too many cells exist to fit within the available space. The form of damage used here is obviously random and distributed across the entire phenotype. Previous work [3] has however shown that there is a similar robust response to localised damage.

5 Discussion

The results from figure 2 consistently show that the embryogeny model produces individuals with remarkable robustness to phenotypic damage, considering that 50% of the phenotype was removed at random. They also show, to varying

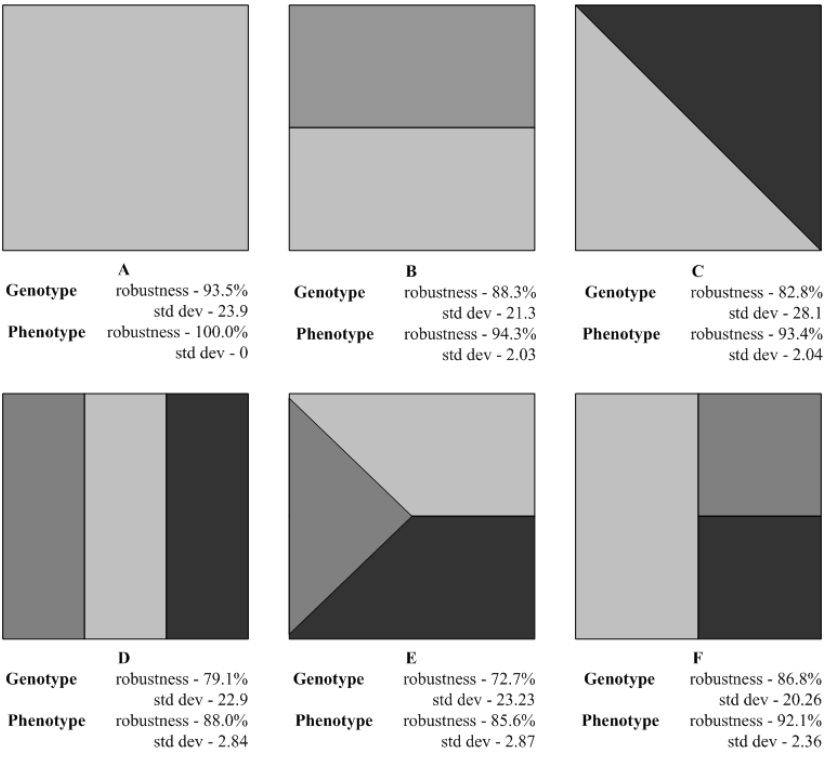


Fig. 2. The various templates used for evaluation and their genotypic and phenotypic robustnesses

degrees, robustness to genetic mutations. However this is hugely dependent upon which gene is mutated as evidenced by the large standard deviations.

Since fitness evaluation is simply a numerical measure of how close an individual fits to a particular template then there is no direct constraint on selection pressure to ensure robustness is evolved. It is also true that the genetic representation can support mappings which result in less robust individuals. Therefore these robust characteristics must be a direct consequence of some bias inherent in the embryogeny representation. This raises the interesting questions of why these abilities are displayed and why they vary across different target templates.

The introduction of such a complex mapping into a simulated evolutionary framework allows for the potential of individual genes to interact with multiple aspects of the phenotype, a principle termed pleiotropy [2]. It is reasonable to assume in this case, that the genetic representation enables some combination of genes in the genome to result in some specific structure in the phenotype. It is also reasonable to assume that if the interaction of these genes with other genes in the genome is high, then mutations elsewhere in the genome are more likely to be destructive to the phenotypic structure for which these genes are useful and hence result in higher levels of pleiotropy. Therefore, it can be argued

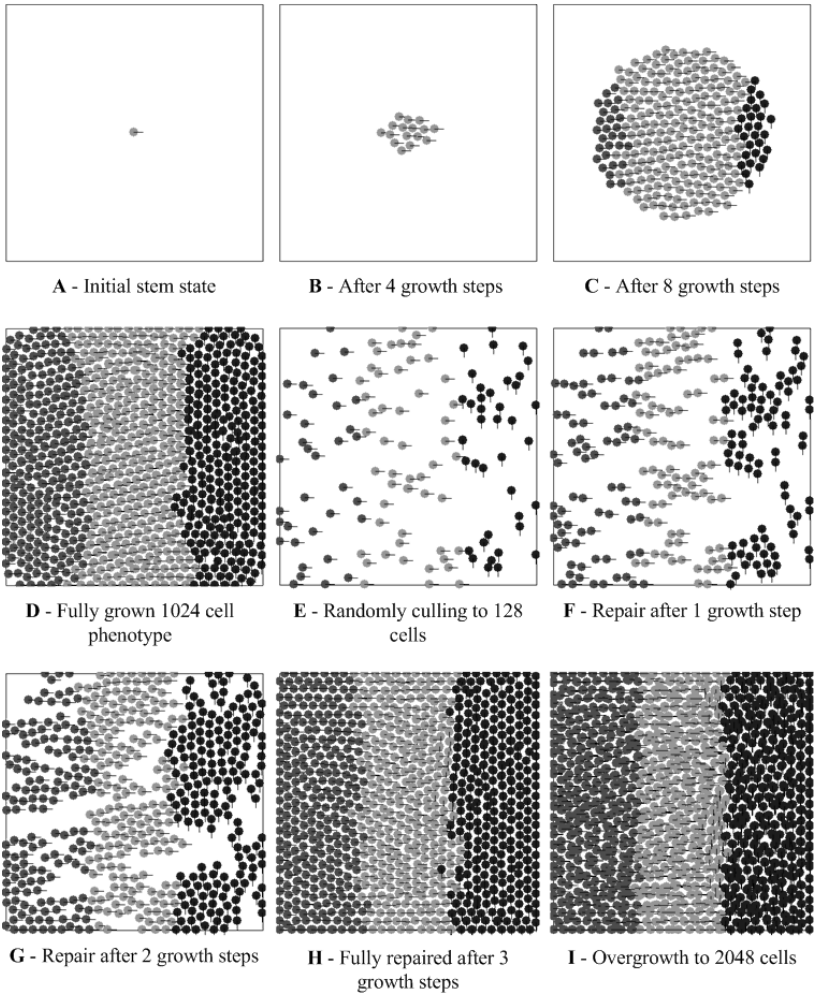


Fig. 3. The typical behaviour of a phenotype in response to damage

that genetic structures with lower levels of pleiotropy should be more likely to result in viable offspring when subjected to search operators. We would expect, given the evolutionary regime, that during both adaptive and neutral evolution selection will cause such a bias towards genes that not only contribute to fitness but which are more likely to stay fit when subjected to search operators.

Figure 4 shows a measure of pleiotropy averaged over 10 evolutionary runs for the template shown in figure 2D. It is measured as the average dependence of various traits in the phenotype on specific genes in the genome. This dependence is considered to be the average percentage change in the distinct characteristics of a phenotype as a result of the removal of an individual gene. This is calculated by removing individual genes from the genomes in turn and then observing the

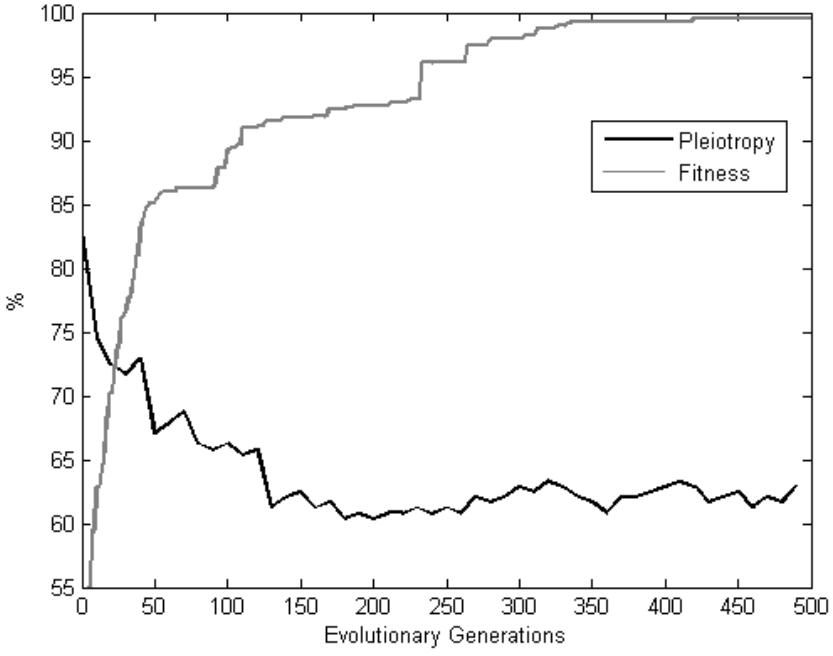


Fig. 4. Average dependence of phenotype and fitness on individual genes

changes in the phenotype. If the changes are large amongst a number of phenotypic characteristics then the pleiotropy will be high whilst small changes across fewer phenotypic characteristics will result in a smaller measure of pleiotropy. It is clear that, as evolution progresses, this measure of pleiotropy decreases and then levels out once the optimal solution is found.

The results of figure 4 suggest that evolved genomes tend to consist of sets of genes exhibiting low levels of pleiotropy, which build specific structures in the phenotype that are identifiable through the target template e.g. areas of colour or shape. This effect can be seen in genomes evolved in this work. For example, genomes tend to have independent sets of genes controlling cell differentiation to each of the colours contained in the phenotype.

Looking at the results across each of the templates from figure 2, it appears that the robustness qualities seem to reduce as the pattern structure becomes more complex. Since fewer genes will be required to express simpler patterns then their genomes have potential for greater redundancy.

6 Conclusion

This paper has demonstrated the integration of an embryogeny mapping process into an evolutionary framework. It has been argued that the mapping process itself has specific impacts upon how evolution will perform, more specifically

maintaining low levels of pleiotropy. This characteristic of the genomes results in phenotypes which are more robust to noise since changes in the environmental state only impact on those relevant parts of the genome and hence relevant parts of the phenotypic structure, which makes the phenotype more resistant to noise and thus more capable of self-regulation and self-maintenance.

The evolutionary bias towards low pleiotropy is dependent on how sets of genes in the genome form parts of the structure in the phenotype. For more complex phenotypes, a greater number of constructional gene sets may be required, with greater levels of interaction, in order to express a more complex shape. It is likely that the impact of this bias towards lower pleiotropy will have a diminishing impact on the overall ability of a phenotype to be robust to noise and damage.

Most importantly, this work has demonstrated a direct relationship between phenotypic robustness and an embryogeny mapping approach. These kind of characteristics may prove to be useful for research areas such as evolvable hardware [12] and artificial neural networks [14]. Indeed recent work in both these areas has shown promising results [4,7]

Future work is to try to produce a comparative model which does not bias towards low pleiotropy. This would enable the analysis of the statistical significance of the robustness observed in the results presented here. At this stage it is not clear how this can be done without irrevocably damaging the model.

Acknowledgments

This work is supported by the EPSRC through a Doctoral Training Account and the School of Computer Science at the University of Birmingham. The author wishes to express gratitude for the help and supervision provided by Dr John A. Bullinaria and for original inspiration and guidance provided by Dr Julian F. Miller.

References

1. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland, 3rd edition, 1994.
2. L. Altenberg. Genome Growth and the Evolution of the Genotype-Phenotype Map. *Lecture Notes in Computer Science*, 899:205–259, 1995.
3. C. P. Bowers. Evolving Robust Solutions with a Computational Model of Embryogeny. In J. M. Rossiter and T.P. Martin, editors, *Proceedings of the UK Workshop on Computational Intelligence: UKCI'2003*, pages 181–188, Bristol, UK, 2003. University of Bristol.
4. C. P. Bowers and J. A. Bullinaria. Embryological Modelling of the Evolution of Neural Architecture. In A. Cangelosi, G. Bugmann, and R. Borisyuk, editors, *Proceedings of the Ninth Neural Computational and Psychology Workshop*. World Scientific, 2004.
5. I. Harvey. Artificial Evolution for Real World Problems. In T. Gomi, editor, *Evolutionary Robotics: From Intelligent Robots to Artificial Life: ER'97*, pages 127–149. AAI Books, 1997.

6. S. Kumar and P. Bentley. Computational Embryogeny: Past, Present and Future. In Ghosh and Tsutsui, editors, *Advances in Evolutionary Computing, Theory and Applications*, pages 461–478. Springer, 2003.
7. H. Liu, J. F. Miller, and A. M. Tyrell. An intrinsic robust transient fault-tolerant development model for digital systems. In J. Miller, editor, *WORLDS Workshop, The Genetic and Evolutionary Computation Conference: GECCO'2004*, 2004.
8. D. Mange, M. Sipper, and P. Marchal. Embryonic electronics. *Biosystems*, 51(3):145–152, 1999.
9. J. F. Miller and W. Banzhaf. Evolving the Program for a Cell: From French Flags to Boolean Circuits. In S. Kumar and P. Bentley, editors, *On Growth, Form and Computers*. Academic Press, London, UK, 2003.
10. M. Nowostawski and R. Poli. Parallel Genetic Algorithm Taxonomy. In L. C. Jain, editor, *Proceeding of the Third International Conference on Knowledge Based Intelligent Information Engineering Systems: KES'99*, pages 88–92, Adalaide, 1999. IEEE.
11. C. Ortega, D. Mange, S. L. Smith, and A. M. Tyrrell. Embryonics: A bio-inspired cellular architecture with fault-tolerant properties. *Genetic Programming and Evolvable Machines*, 1(3):187–215, 2000.
12. V. K. Vassilev and J. F. Miller. Scalability Problems of Digital Circuit Evolution. In D. Keymeulen J. Lohn, A. Stoica and S. Colombano, editors, *Proceedings of the 2nd NASA/DOD Workshop on Evolvable Hardware*, pages 55–64, Los Alamitos, CA, 2000. IEEE Computer Society.
13. L. Verlet. Computer Experiments on Classical Fluids I. Thermodynamical Properties of Lennard-Jones Molecules. *Phys. Rev.*, 159(1):98–103, 1967.
14. X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

Topology Changes Enable Reaction-Diffusion to Generate Forms

Shuhei Miyashita[1] and Satoshi Murata[2]

[1] Artificial Intelligence Laboratory, Department of Informatics, University of Zurich, Switzerland
miya@ifi.unizh.ch
[2] Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Japan
murata@dis.titech.ac.jp

Abstract. This paper demonstrates some examples that show the ability of reaction-diffusion mechanism to code the curvature of forms of multi-cellular systems. The simulation model consists of two layers: the first generates reaction-diffusion waves and the second diffuses chemical substances. The results show that topology changes feedback information to the reaction-diffusion mechanism allowing the control of the morphogenetic process.

Keywords: morphogenesis, reaction-diffusion, geometric topology.

1 Introduction

Multicellular organisms usually consist of a large number of cells, which are able to form the shape of an organism by an intricate web of cell-cell interactions, a process called morphogenesis. As each cell contains the same genome, this feat is realized in a distributed and autonomous way with the absence of any centralized control. Although the elucidation of the molecular mechanisms made big progress in biology, an overall picture is still lacking.

In this paper, we hypothesize that morphogenesis depends on the following two conditions:

1. Chemical substances (morphogens) play a role in encoding directly morphological information. In this paper we hypothesize that some substances transmit information by its concentration.
2. Morphogenesis is an autonomous, distributed process without any centralized control for all cells.

We used these two conditions as guidelines to screen the existing literature of morphogenetic models. Alan Turing's reaction-diffusion model [1] uses two chemical substances able to produce spatial patterns in space. As this model uses gradients the first condition is fulfilled, but was not used to form shapes. Essentially reaction-diffusion mechanisms are means of breaking the symmetry among homogenous cells in autonomous and distributed way and therefore

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 159–168, 2005.

it also fulfills the second condition. L.Wolpert suggested the concept of positional information enabling the cells to know where they are [2]. Gierer and Meinhardt [3] used Turing's model for pattern formations. Murray [4,5] also presented a possible mechanism for pattern formation in animal markings using reaction-diffusion. Crampin [6] explored the patterns of reaction-diffusion wave accompanied by the extension of space. Kondo [7] pointed out that the change of the stripe pattern in angelfish is driven by reaction-diffusion mechanism. C.Furusawa and K.Kaneko [14] found the phenomena of dynamical cell differentiation by creating their own model. However, all the above examples are mainly focused on pattern generation and not on shape forming. There exist many approaches for morphogenesis - especially in the field of Artificial Life - which can be divided into several types: Lindenmayer grammars [11], cellular automata [12,18], strictly mechanical approaches where physical interactions between the cells were programmed to simulate morphological processes [10], recurrent diagram networks to express the bodies of simulated creatures [13]. However, the correspondence between these models and real organisms has been considered less seriously. More recently there was renewal of interest in the relations between gene regulatory networks and morphology [15][17][16][19][20], but these models pay little attention to the relation between morphogenesis and reaction-diffusion mechanism. The reason may be that only a few people noticed a possible link between pattern generation and morphological form of creatures. In this paper the linking between reaction-diffusion mechanisms and cell division and physical interactions between the cells can be used to produce shapes of organisms. We focused our research on the change of the geometric topology of cellular networks and found that topology changes can be used to feedback information from the transformed field to the reaction-diffusion mechanism. This feedback made it possible to create a model of the gastro-intestinal tract as an example to show how each homogeneous cell realizes global shapes by computer simulation. The main point of this paper is that the feedback of information of topological changes about the reaction-diffusion mechanism is an essential ingredient to model morphogenetic processes by reaction-diffusion approaches.

This paper is constituted as follows: First biological background is explained, reaction-diffusion system and the developed model are presented in the next section 2. In the third section, the simulation results are shown and the fourth section discusses the results and the conclusions are presented in the last section.

2 Model

In general, multicellular organisms, especially animals, have a gastrointestinal tract, which is essentially a tube from mouth to anus. A cross section of the gastrointestinal tract in humans can be divided into three layers. Going from the inside to outside, the first layer is the epithelium that covers the surface with epithelial cells, connective tissue, and a muscular coat that takes on a role of contraction [8]. Epithelial cells are connected to each other through tight junctions, adhesion belts by cadherins and desmosomes, and gap junctions through which small molecules can pass. Epithelial cells connect to a matrix below (Extra-

Cellular-matrix:ECM) via hemidesmosomes or integrins. For a long time, the ECM had been considered as a physical crutch or anchorage. Recently, however, it became clear that the ECM has more active functions such as passing some specific molecules or controlling the form of a cell that is attached to it. Taking the human gastrointestinal tract as an example, three kinds of curvatures of different scales on epithelium layer can be observed. They are called plicae circulares, villi and microvilli in descending order. This architecture increases the surface area, which facilitates the uptake of food by the gastrointestinal tract. Focusing on this hierarchical form of the epithelial surface, H.Honda advocates that in general the form of multi cellular system is realized as two-dimensional sheets rather than three-dimensional solids [9].

2.1 Two Layer Reaction-Diffusion Model

The cross-section of the gastrointestinal tract can be modeled as two layers for simplification, the epithelial layer and the ECM, which includes submocosal layer and the below. The simulations are performed on a one-dimensional cell array. Fig.1 shows a schema of this model and its correspondence to the biological gastrointestinal tract. Upper nodes represent cells, lower ones represent connection points of each epithelial cells and ECM. Both are expressed as mass points. Links between them are represented as chemical and mechanical connections (chemical substances only diffuse through horizontal and vertical connections, see Fig.1). The mechanical interactions are expressed as spring and damper connections. Epithelial cells contain two chemical substances that react and diffuse, in order to generate reaction-diffusion waves. A reaction-diffusion wave is a periodic spatial concentration pattern (see [1]). The general form of a two chemical reaction-diffusion system can be expressed as partial differential equations (eq.1,2).

$$\dot{u} = f(u, v) + D_u \nabla^2 u \tag{1}$$

$$\dot{v} = g(u, v) + D_v \ \nabla^2 v \tag{2}$$

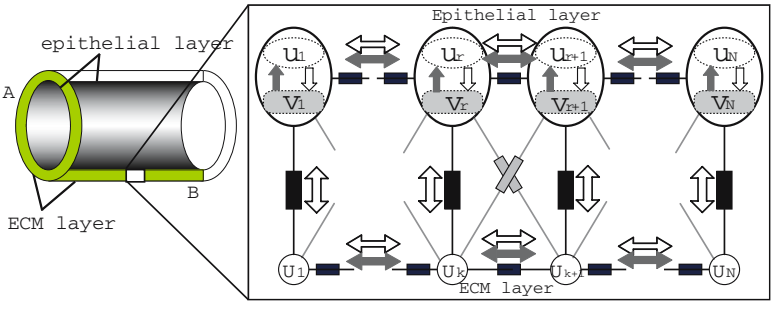


Fig. 1. Two layer model. Upper layer represents epithelial cells and lower layer represents ECM. Epithelial layer can generate reaction-diffusion wave. ECM just diffuses chemical substances.

$$f(u,v) \equiv +5u - 6v + 1\underline{-(u-1)^3}$$
$$= +2u + 3u^2 - u^3 - 6v + 2 \quad (3)$$
$$g(u,v) \equiv +6u - 7v + 1\underline{-(v-1)^3}$$
$$= +6u - 10v + 3v^2 - v^3 + 2 \quad (4)$$
$$\dot{u} = D\nabla^2 u \quad (5)$$

Where u, v represent concentrations of the two chemicals, $f(u,v)$ and $g(u,v)$ represent reaction parts between chemicals u and v, respectively. $\nabla^2 u$ and $\nabla^2 v$ represent the Laplacian of u and v respectively. D_u, D_v and D represent diffusion coefficient of activator, inhibitor and also activator, respectively. Usually a proportion D_v/D_u plays a key role in the behavior of reaction-diffusion system. Since chemical substances diffuse to neighboring cells and the ECM via different channel, it seems reasonable to assume that the diffusion coefficients differ among internal epithelial connections and among epithelial-ECM connections. We set that only the activator can pass through the connection between epithelial cell and ECM. Eq.3,4 is applied for the reaction part, which adds the non-linear term (underlined in eq.3,4) to the Turing's model in order to be more stable fulfilling a definition of activator and inhibitor $\left(\frac{\partial u}{\partial v} < 0, \frac{\partial v}{\partial u} > 0\right)$. The equilibrium points are the same as Turing's $(u = 1.0, v = 1.0)$. The function of the ECM is just to diffuse chemical substances. Its general form can be expressed in eq.5. To the system, Dirichlet boundary conditions were applied, which set the chemical flow at the boundary to zero $\left(\frac{du_1}{dx} = \frac{du_N}{dx} = 0, \frac{dv_1}{dx} = \frac{dv_N}{dx} = 0, \frac{dU_1}{dx} = \frac{dU_N}{dx} = 0\right)$. Here, $u_1, u_N, v_1, v_N, U_1, U_N$ represent the concentrations of boundary cells and ECM.

2.2 Cell Cycle and Cell Division

Cell cycle is determined by various factors, in this paper, the condition for cell division depends on the concentration and a specific threshold. Cells divide when the concentration of the activator is kept over a specific threshold for a certain time. The concentration of each chemical substance right before division is applied to the concentration of the cell divided.

Fig.2 illustrates the rule controlling how cells reconnect after they divided. Where a represents distance from epithelial cell to ECM and b represents connection length between two cells. Fig.2 a) shows that after each cell divides, a new

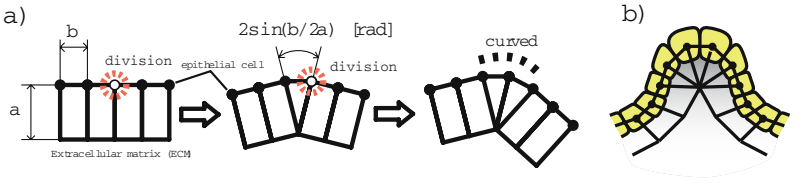


Fig. 2. Reconnection by cell division. a)After each cell divides, a new link is added, then an $2sin(b/2a)$[rad] angle of curvature is created depending on the ratio of the length of horizontal and vertical links. b)Correspondence to real tissue.

link is added, then an $2sin(b/2a)$[rad] angle of curvature is created depending on the ratio of the length of horizontal and vertical links. Since the operating mechanism of the extend speed of ECM hasn't been clear, the ECM's extension speed is assumed to be constant. Fig.2 b) illustrates the correspondence to real tissue after the curvature is created.

3 Simulations

$$\begin{aligned} du_r/dt &= +2u_r(t) + 3u_r^2(t) - u_r^3(t) - 6v_r(t) + 2 \\ &\quad + D_{u_{epi}}\{u_{r-1}(t) + u_{r+1}(t) - 2u_r(t)\} + D_{u_{epi\ \ ECM}}(U_k - u_r) \quad (6) \\ dv_r/dt &= +6u_r(t) - 10v_r + 3v_r^2(t) - v_r^3(t) + 2 \\ &\quad + D_{v_{epi}}\{v_{r-1}(t) + v_{r+1}(t) - 2v_r(t)\} \quad (7) \\ dU_k/dt &= D_{u_{epi\ \ ECM}} \sum_r (u_r - U_k) + D_{u_{ECM}}(U_{k-1} + U_{k+1} - 2U_k) \quad (8) \end{aligned}$$

By discretizing the space the eq.6,7 can be obtained from eq.3,4. In the same way, eq.8 can be obtained from eq.5. These equations are used to calculate the dynamics of the chemicals, where, u_r and v_r represents the concentration of activator and inhibitor in epithelium cell, respectively. U_k represents the concentration of activator in ECM. Subscript r and k is an identification number of the cell and the ECM, respectively. The parameters are set as follows: $D_{u_{epi}} = D_{u_{ECM}} = 1.0, D_{v_{epi}} = 3.0, D_{u_{epi\ \ ECM}} = 0.53, u_{threshold} = 1.035$, initial concentration $(u, v) = (1.0, 1.0)$(equilibrium point). If the concentration exceeds a division threshold for more than 50 steps (step is defined below), a cell divides. Initial perturbations are always added at the center of the epithelial layer.

$$m\ddot{q_i} + c\sum_j(\dot{q_i} - \dot{q_j}) + k\sum_j\left(1 - \frac{l}{|q_i - q_j|}\right)(q_i - q_j) = 0 \quad (9)$$

Cells move on a two-dimensional space. The equation of motion for the cell i is expressed in eq.9 where subscript i, j is an identification number of the cell or connection point to ECM. The position of the cell q_i is defined as a vector. The cell which exists in neighbor of cell i is denoted as j. m, c, k, and l denote mass, damper coefficient, spring coefficient and natural length of spring, respectively. These parameters are set as follows: $m = 9, c = 30, k = 50, l_{hor} = 18, l_{ver} = 40$. Every differential equation is integrated by the Euler method ($\delta t = 0.05$, 1step=$20\delta t$). Mechanical forces are calculated assuming in vivo time scales ($\delta t = 0.005$, 1step=$60\delta t$).

Fig.3 a),b), and c) show the simulation results in chronological order. The number of initial epithelial cells are set to 8,64 and 128, respectively. White colored cells represent those in which the concentrations of activator exceed the threshold. Fig.3 a) shows a sectional side view of the gastrointestinal tract (Fig.1 B). It can be seen that the curvature is created depending on the difference of the extension speed between epithelial layer and ECM. Fig.3 b) shows the

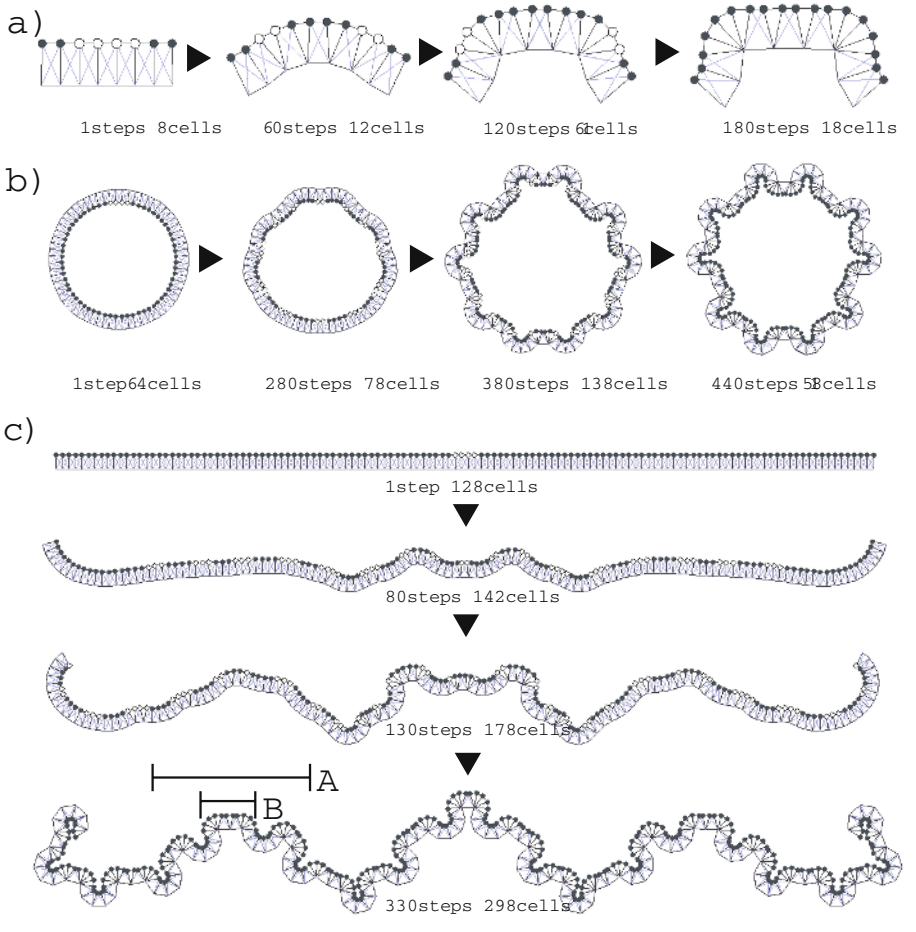


Fig. 3. Simulation results in chronological order. a)Sectional side view of the gastrointestinal tract (Fig.1 B). It can be seen that layers bend by cell division. b)The longitudinal section of the gastrointestinal tract (Fig.1 A). Curvatures are also created independently of the boundary condition. c)Sectional side view of the gastrointestinal tract (Fig.1 B). Unexpectedly, two different curvatures of different scales are formed like Koch-curve. In all results, epithelial cells stop dividing automatically at the end.

longitudinal section of the gastrointestinal tract. The internal layer corresponds to the epithelial layer and the external layer to the ECM (Fig.1 A). Perturbation is added on the top. Curvatures are also created independently of the boundary condition. Fig.3 c) shows a sectional side view of gastrointestinal tract (Fig.1 B). As time passes, unexpectedly, two different curvatures of different scales are formed on the surface. Finally, fractal hierarchies like Koch-curve are observed (We shall return to this point later). In all results, it can be seen that epithelial cells stop dividing automatically at the end.

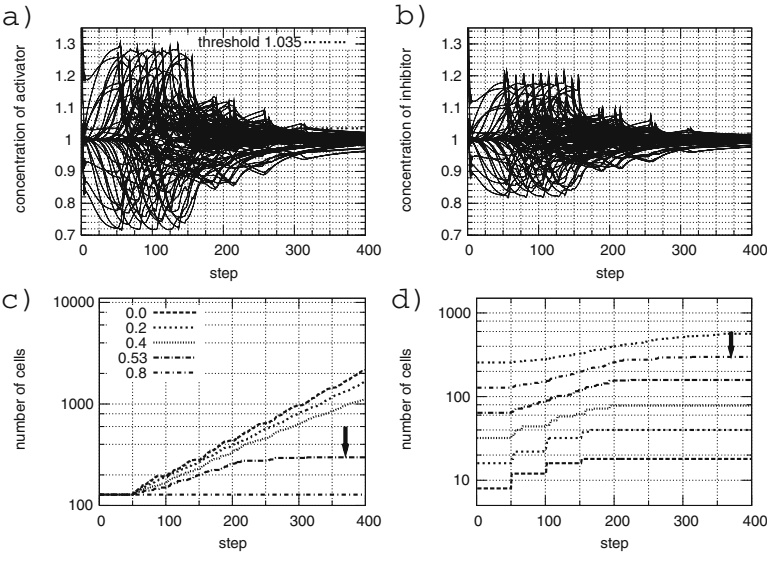


Fig. 4. Transition graph. a)Transition of concentration of activator. b)Inhibitor. These concentrations converge as time passes and no concentration of activator exceeds division threshold after 329 steps. c)d)Increase of the number of cells. c)To quantify the role of the ECM, the diffusion coefficient between the epithelial layer and the ECM is changed from 0.0 to 0.8 and plotted on it. As the figure shows the smaller the coefficient is, the faster the cells grow. d)Only the initial number of cells differs. It is revealed that the mechanism of growth convergence is not dependent on the initial number.

Fig.4 a) and b) show the concentration transition of all cells that start at 128 cells. a) and b) show concentration of activator with division threshold and inhibitor, respectively. X-axis represents time and Y represents concentration. As time passes, these concentrations converge and none of the concentrations of activator exceed division threshold after 329 steps. c) and d) show the increase of cell number. The X-axis represents time and Y-axis number of cells in logarithmic

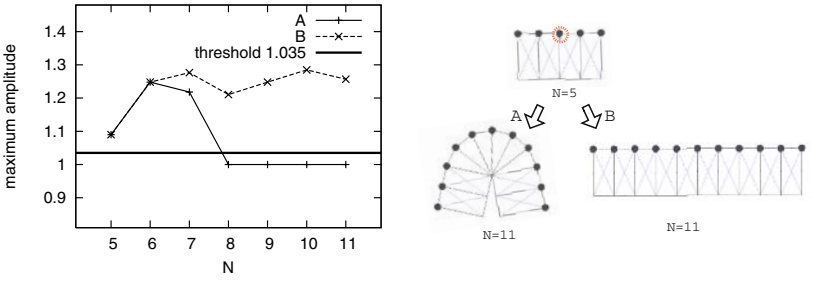


Fig. 5. Maximal amplitude of the activator's concentration. Increase of the curvature suppresses reaction-diffusion wave.

scale. (lines pointed by arrows in c) and d) show this model). To quantify the role of the ECM, the diffusion coefficient between the epithelial layer and the ECM changed from 0.0 to 0.8 and plotted on c). The figure shows the smaller the coefficient is, the faster the cell grows. When the diffusion coefficient is 0.8, the number of cells doesn't change, because none of the concentrations of activator exceed threshold. This indicates that the diffusion coefficient between epithelial layer and ECM influences the speed of cell growth. Initial number of cells is changed and plotted in d). It can be seen that the growth mechanism converges independently of the initial number of the cells. The growth rate doesn't change after each transition (around 220% to 250%).

4 Discussions

4.1 The Suppression of Amplitude of Reaction-Diffusion Wave

Fig.5 shows the maximal amplitude of the activator's concentration generated by the epithelial layer after the number of cells increased from 5 to 11 in different cell topologies (A and B). The X-axis represents the number of cells and the Y-axis the largest concentration. It can be seen that the increase of the curvature suppresses the reaction-diffusion wave. When the number of cells becomes larger than 8, all concentrations of activator become below threshold. This is because the ECM, which connects several cells plays a role of averaging activator that is contained in epithelial cell (Eq.8). Thus as the number of connecting cells increases, concentration of ECM becomes closer to the average of all concentrations that connect (usually $u = 1.0$). Consequently, the reaction-diffusion wave is inhibited. This can also be confirmed by settling all concentration of activator of ECM 1.0 when layers are straight.

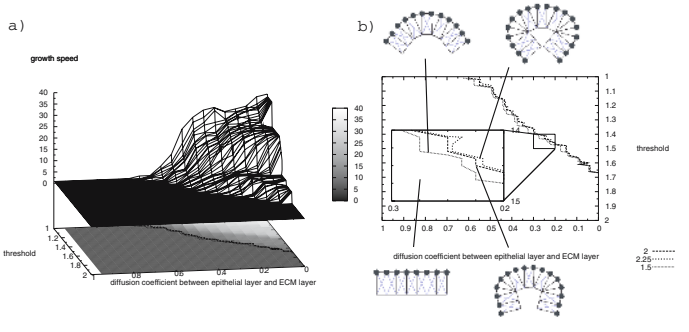


Fig. 6. Growth speed of epithelial cells depending on two parameters. a) The X-axis represents diffusion coefficient, Y represents division threshold and Z represents growth speed. b) X-Y plane area of the same figure. Though it seems these parameters have thin range, note that the curvature depends on the ratio of horizontal-vertical length of the links (see Fig.2).

Fig.6 shows the growth speed of epithelial cells depending on two parameters: the division threshold and the diffusion coefficient between epithelial layer and ECM. The growth speed is defined as follows: the number of cells at 300 steps/initial number of cells (this time 8). In fig.6 a), X-axis represents diffusion coefficient, y division threshold and z growth speed. In fig.6 b), X-Y plane area is also represented with contours and its forms when the growing speed is 1.5, 2.0, and 2.25. By shifting diffusion coefficient to smaller, the ratio of reaction-parts/diffusion-parts gets lager in their reaction-diffusion system. This makes the amplitude of reaction-diffusion wave bigger: the smaller the diffusion coefficient becomes, the faster growth speed becomes.

4.2 Hierarchy of the Form

In Fig. 3 c), two different scales of curvatures are formed (A and B). The small curvatures are formed because the epithelial layer and the ECM have different extending speeds. The large scales are created due to mechanical interactions between cells, which extend initial small mechanical perturbations due to cell divisions. This large curvature is always observed, when the layers have a certain length. Each curvature is formed based on their mechanisms and this is the reason why the scales vary. Since friction between gastrointestinal organs and its outer has not been modeled, external forces can erase the large curvature.

5 Conclusions

This paper focused on geometric topology changes of cell networks and shows that topology changes enable reaction-diffusion mechanisms to control morphologic process. In other words, the shape that is driven by reaction-diffusion can build up a feedback loop back to reaction-diffusion mechanism. The model is comprised of two layers: one is the epithelial layer that generates reaction diffusion waves and the other is the ECM that diffuses chemical substances. Cells divide depending on the reaction-diffusion mechanism. Once the shape gets round, chemical substances are averaged at middle point, inhibiting the amplitude of the reaction-diffusion wave. Since the cell division depends on the concentration of activator in this model, it restricts system's growth. This closed feedback loop is one of the autonomous distributed ways to code morphogenesis by supposing that chemical substances act as a transmitters of positional information.

Acknowledgment

Many thanks to Peter Eggenberger Hotz for helpful suggestions. This research is supported by the Swiss National Science Foundation, project #200021-105634/1.

References

1. A.M.Turing: A reaction-diffusion model for development, The chemical basis of morphogenesis, Phil Trans. Roy. Soc. of London, Series B; Biological Sciences, **237**, 5,PP.37-72 (1952)
2. L.Wolpert: Positional Information and the Spatial Pattern of Cellular Differentiation, J. Theoret. Biol., **25**, PP.1-47 (1969)
3. A.Giere, H.Meinhardt: A theory of biological pattern formation. Kybernetik, **12**, PP. 30-39 (1972)
4. J.D.Murray: A pattern formation mechanism and its application to mammalian coat markings. In 'Vito Volterra' Symposium on Mathematical Models in Biology, **39**, PP. 360-399 (1979)
5. J.D.Murray: On pattern formation mechanisms for Lepidopteran wing patterns and mammalian coat markings, Phil. Trans. Roy. Soc. of London B, **295**, PP. 473-496 (1981a)
6. E.J.Crampin, E.A.Gaffney, and P.K.Maini: Reaction diffusion on growing domains: scenarios for robust pattern formation, Bull. Mathematical Biology, **61**, PP.1093-1120 (1999)
7. S.Kondo: A reaction-diffusion wave on the skin of the marine angelfish Pomacanthus, Nature, **376**, PP. 765-768 (1995)
8. B.Alberts, A.Johnson, J.Lewis, M.Raff, K.Roberts, P.Walter: Molecular biology of the cell. 4th ed. Garland Science (2002)
9. H.Honda: The mathematical and physical analysis of morphogenesis, Kyoritsu-shuppan (2000)
10. M.Odell, G.Oster, P.Alberch: The Mechanical Basis of Morphogenesis, Developmental Biology **85**, PP.446-462 (1981)
11. A. Lindenmayer: Mathematical Models for Cellular Interaction in Development, Journal of theo-retical Biology, **18**, PP.280-300 (1968)
12. C.G.Langton: Self-reproduction in cellular auto-mata, Physica D: Nonlinear Phenomena, **10**, Issues 1-2, PP.135-144 (1984)
13. K.Sims: Evolving 3d morphology and behavior by competition. Artificial Life, **1**, Issue 4, PP.353-372 (1995)
14. C.Furusawa and K.Kaneko: Emergence of Multicellular Organisms with Dynamic Differentiation and Spatial Pattern, Artificial Life 4 PP.79-93 (1998)
15. P.Eggenberger: Evolving morphologies of simulated 3d organisms based on differential gene expression, Fourth European Conference of Artificial Life, Cambiedge, MIT Press (1997).
16. P.Eggenberger: Genome-pysics interaction as a new concept to reduce the number of genetic parameters in artificial evolution. Proceedings of the Congress of Evolutionary Computation, Canberra (2004)
17. J.Bongard: Evolving modular genetic regulatory networks. Proceedings of The IEEE 2002 Congress on Evolutionary Computation PP.1872-1877 (2002)
18. K.Tomita, H.Kurokawa, S. Murata: Graph Auto-mata: Natural Expression of Self-reproduction, Physica D: Nonlinear Phenomena, **171** No.4, PP.197-210 (2002).
19. K.Fleischer and A.H.Barr: A simulation testbed for the study of multicellular development : The multiple mechanisms of morphogenesis, Proceedings of the Workshop on Artificial Life, Santafe, NewMexico, PP.389-416 (1994)
20. D.Stork, B.Jackson and S.Walker: Non-Optimality via Pre-adaptation in Simple Neural Systems, Artificial Life II, Addison-Wesley (1992)

Aggregation Behaviour as a Source of Collective Decision in a Group of Cockroach-Like-Robots*

Simon Garnier[1], Christian Jost[1], Raphaël Jeanson[1], Jacques Gautrais[1], Masoud Asadpour[2], Gilles Caprari[2], and Guy Theraulaz[1]

[1] Centre de Recherches sur la Cognition Animale, UMR-CNRS 5169, Université Paul Sabatier, 118 route de Narbonne, F-31062 Toulouse cedex 4, France
simon.garnier@cict.fr
[2] Autonomous Systems Lab, Swiss Federal Institute of Technology, (EPFL), CH-1015 Lausanne, Switzerland

Abstract. In group-living animals, aggregation favours interactions and information exchanges between individuals, and thus allows the emergence of complex collective behaviors. In previous works, a model of a self-enhanced aggregation was deduced from experiments with the cockroach *Blattella germanica*. In the present work, this model was implemented in micro-robots Alice and successfully reproduced the agregation dynamics observed in a group of cockroaches. We showed that this aggregation process, based on a small set of simple behavioral rules of interaction, can be used by the group of robots to select collectively an aggregation site among two identical or different shelters. Moreover, we showed that the aggregation mechanism allows the robots as a group to "estimate" the size of each shelter during the collective decision-making process, a capacity which is not explicitly coded at the individual level.

1 Introduction

Since the last 15 years, collective robotics has undergone a considerable development [18]. In order to control the behavior of a group of robots, collective robotics was often inspired by the collective abilities demonstrated by social insects [3,15]. Indeed, nature has already developed many strategies that solve collective problems through the decentralized organisation and coordination of many autonomous agents by self-organized mechanisms [4].

Among all these self-organized behaviours, aggregation is one of the simplest. But it is also one of the most useful. Indeed, aggregation is a step towards much more complex collective behaviours because it favours interactions and information exchanges among individuals, leading to the emergence of complex

* This work was partly supported by a European community grant given to the Leurre project under the IST Programme (2002-2005), contract FET-OPEN-IST-2001-35506 of the Future and Emerging Technologies arm and by the Programme Cognitique from the French Ministry of Scientific Research. The authors would like to thank Jean-Louis Deneubourg for all its very helpful advices about this work.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 169–178, 2005.

and functional self-organized collective behaviours (for some examples, see [4]). As such it plays a keyrole in the evolution of cooperation in animal societies [6].

Such self-organized aggregation processes were regularly used in collective robotics. For instance, foraging tasks (i.e. clustering of objects scattered in the environment) were used to study the impact of the group size [12] or of a simple form of communication [17] on the harvest efficiency. But even more complex consequences of aggregation processes were studied with groups of robots. For instance, [1] showed that division of labor can emerge in a group of foraging robots when the size of the group grows. [8] showed that an object clustering paradigm based on stigmergy [7] can lead a group of robots to order and assemble objects of two different types.

In this paper we address a new collective behavior that is based on self-organized aggregation of robots themselves. We show that a self-enhanced aggregation process, which leads groups of cockroaches to a quick and strong aggregation [10], can be used by a group of mini-robots Alice to select collectively an aggregation site among two identical or different shelters. We show that, even though these robots have limited sensory and cognitive abilities, they are still able to perform a collective decision. It has already been shown that such self-enhanced mechanisms are used by insects to make collective decisions: for instance in food source selection in bees [16] or in resting site selection in cockroaches [2]. These collective choices appear through the amplification of small fluctuations in the use of two (or more) targets.

We first describe the biological model of aggregation we have used and the way this model was implemented in a group of mini-robots Alice. We then show that this implementation indeed results in a collective aggregation behavior that is quantitatively indistinguishable from cockroach aggregation. Finally, we show that, when this aggregation behavior is restricted to certain zones in the environment (for instance by natural preferences for dark places as in cockroaches [14]), the robots preferentially aggregate in only one of these zones, i.e. they collectively choose a single "rest" site. When these zones are of different sizes, the robots preferentially choose the biggest of the two, but without being individually able to measure their size. The results of our experiments were also used to calibrate a computer simulation model of Alice robots that will allow us to extend the exploration of this collective decision model in further studies.

2 Self-organized Aggregation

The aggregation process cited above is directly inspired by a biological model of displacement and aggregation developed from experiments with first instar larvae of the german cockroach *Blattella germanica* [9,10]. This model was built by quantifying individual behaviors of cockroaches, that is their displacement, the interactions among individuals and with the environment in a homogeneous circular arena (11 cm diameter) . Each of these individual behaviors was described in a probabilistic way: we measured experimentally the probability rate for a given behavior to happen.

This analysis showed that cockroaches display a correlated random walk (constant rate to change direction and forward oriented distribution of turning angles) in the center of the arena [9]. When reaching the periphery of the arena, cockroaches display a wall following behavior (thigmotactic behavior) with a constant rate to leave the edge and return into the central part of the arena [9]. In addition, cockroaches can stop moving at any moment, stay motionless for some time and then move again. Analysis showed that the stopping rate for an individual increases with the number of stopped cockroaches in the direct neighbourhood (within the range of antenna contact) [10]. On the other side, the rate to leave an aggregate decreases with this number [10]. Thus, this dual positive feedback leads to the quick and strong formation of aggregates (see Fig. 1). A more detailed description of the model can be found in [9,10].

The first part of our work was to implement this biological model of aggregation in the micro-robots Alice. These robots were designed at the EPFL (Lausanne, Switzerland) [5]. They are very small robots (22mm x 21mm x 20mm) equipped with two watch motors with wheels and tires allowing a maximum speed of 40 mm s^{-1}. Four infra-red sensors are used for obstacle detection and local communication among Alices (up to 4 cm distance). Robots have a microcontroller PIC16LF877 with 8K Flash EEPROM memory, 368 bytes RAM but no built-in float operations. To determine the number of neighbors (upon which the aggregation process relies), each robot owns a specific identification number and counts the number of nearby neighbors in a distance roughly less than 4 cm. Intrinsic differences between the perception area of robots and cockroaches and imperfect neighbor counts due to noise in IR devices required some fine-tuning of the behavioral parameters in order for the behavioural output of the robots to correctly match the cockroach individual behaviors. This behavioral output of robots was measured using the same experimental methods (10 to 30 experiments depending on the studied behavior) as those used to characterize the individual behavior of cockroaches [9,10].

However the true validation of the model implementation must be done at the collective level by comparing the aggregation behavior of robots to the aggregation behavior of cockroaches. To this aim, we ran the following aggregation experiment: groups of robots (10 or 20 individuals) were put into a homogeneous white circular arena (50 cm diameter) during 60 minutes. This experiment is similar to the one done by [10] with cockroaches. To draw a parallel between cockroach aggregation behavior and robot aggregation behavior, we scaled the dimensions of the arena so that it matches scale differences between robot and cockroach sizes. The experiment was repeated 10 times for each group size. The aggregation dynamics were characterized through three kinds of measurements (sampled every minute): size of the largest aggregate, number of aggregates and number of isolated individuals (see [9,10] for a detailed description of these measurements). The experimental results showed a very good agreement between robots and cockroaches, confirming that the cockroach aggregation process was well implemented in the Alice robots (see Fig. 1).

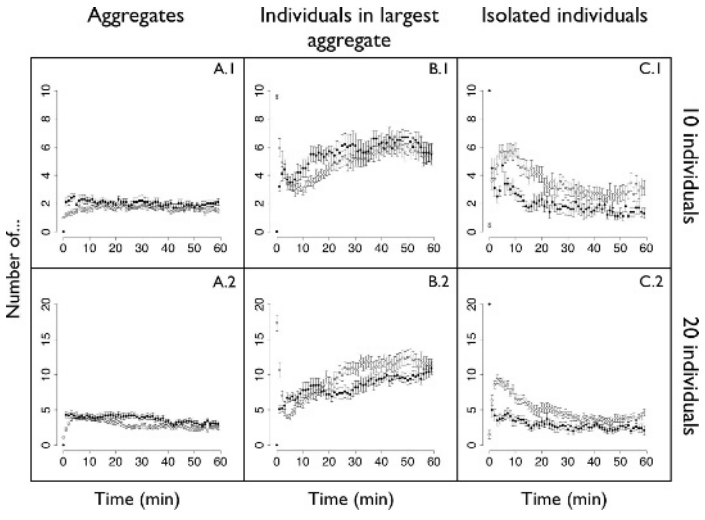


Fig. 1. Aggregation dynamics.**A**: number of aggregates. **B**: size of the largest aggregate. **C**: number of isolated individuals. **1**: experiments with 10 individuals. **2**: experiments with 20 individuals. Black dots represent data for robots; white dots represent data for cockroaches. Each dot represents the mean ± standard error (s.e.). Initial differences between starting points of robot and cockroach dynamics are solely due to the way cockroaches have to be brought into the arena as explained in [10].

3 Collective Choice

This aggregation process implemented in robots can occur anywhere in the whole experimental arena, with no preference for a given location. Actually, in nature some places are more attractive for cockroaches, thus promoting aggregation in particular sites. For instance, cockroaches preferentially aggregate in dark places [14]. Experimentally, if one puts a dark shelter in a lighted arena (as the one used for the study of cockroach aggregation), one can observe that cockroaches strongly aggregate under this shelter. And if two or more dark shelters are placed in the arena, one can observe that a majority of cockroaches aggregates under only one of these shelters, rather than spreading evenly among all the aggregation sites [11]. Hence cockroaches are able to perform a collective choice for a given aggregation site, even if these sites are identical.

Though the mechanisms leading to this collective choice are not yet fully understood, we suggest that this choice could strongly rely on the self-enhanced aggregation process described above and tested with robots. In a recent paper, [2] showed that the simple modulation of the resting period on a given site by the number of individuals on that site leads the group of cockroaches to the choice of one shelter among two or more identical ones. We suggest here that this modulation can be implemented easily through the aggregation process described above. To test our hypothesis, we ran three sets of experiments during which a

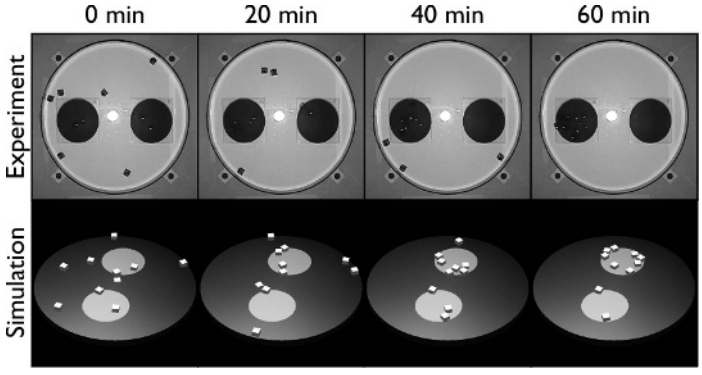


Fig. 2. Snapshots of an experiment (top) and a simulation (bottom) taken every 20 minutes during 60 minutes. These snapshots correspond to the experiment with two identical shelters (14 cm diameter). As can be seen, the experiment ended with the choice of one of the two shelters by both real and simulated robots.

group of robots faced the choice between two potential aggregation sites. Besides proving that a collective decision can appear in robots from a simple aggregation process, these experiments were also used to calibrate a simulation tool which will be used in further studies to identify the behavioral parameters that control collective choice (see Fig. 2 for some pictures of both experiments wih robots and simulations). In the following, all statistical computations will be made in the free software R [13].

The first set of experiments was designed to ascertain whether the cockroach aggregation behavior is able to lead a group of robots to a collective choice between two identical targets. To that aim, we put a group of 10 robots in the same arena as the one used for aggregation experiments, except that we added just above the arena two dark shelters. These shelters were of the same size (14 cm diameter) and each of them can house the whole population of robots. Robots used the same behavioral algorithm as the one previously tested for its aggregation ability, except that, now, robots only stop under dark shelters (that is when IR light intensity falls under a given threshold). 20 experiments were performed, each lasting 60 minutes.

The number of stopped robots under each shelter was measured every minute to characterize the aggregation dynamics under each shelter. In addition, we also computed the percentage of stopped robots under each shelter at the end of each experiment to characterize the collective choice of the group of robots. From this last measurement, we derive what we call a "choice distribution". For a given shelter, this choice distribution corresponds to the number of experiments ending with a given percentage of stopped robots under this shelter (the choice distribution being symmetrical for the other shelter). Note that a robot can be in one of these three locations at the end of an experiment: under shelter 1, under shelter 2 or outside the shelters. In the case of each robot choosing randomly a shelter (i.e. without any influence of its conspecifics), the result will follow a trinomial law

with parameters $m_{tot} = 10$ (number of robots), $p_a = (m_{tot} - m_s)/mtot$ (p_a, probability for a robot to be outside the shelters; m_s number of robots stopped under any shelter, estimated from the experiments), $p_{s1} = (1 - p_a)(r_{s1}^2/(r_{s1}^2 + r_{s2}^2))$ (p_{s1}, probability for a robot to be under shelter 1; r_{s1}, radius of shelter 1; r_{s2}, radius of shelter 2) and $p_{s2} = 1 - p_{s1} - p_a$ (p_{s2}, probability for a robot to be under shelter 2). The choice distribution resulting from this trinomial law can be obtained through Monte Carlo simulations (10000 simulations of 20 replicates). In the case of identical shelters, this choice distribution displays a centered peak as can be seen in Fig. 3 B.1, meaning that a majority of experiments should end with no choice for a particular shelter.

Contrary to the trinomial resulting choice distribution, the choice distribution obtained in experiments with two identical shelters displays two peaks, one at each side (see Fig. 3 B.2). A chi square test shows a strong difference between the trinomial and experimental distributions ($\chi^2 = 367.7$, $df = 4$, $p < 0.0001$). Similar results are obtained with simulations (see Fig. 3 B.3) and a chi square analysis of contingency tables shows no difference between experiments and simulations ($\chi^2 = 2.1$, $p = 0.7322$, p-value simulated with 10000 replicates). This U-shape distribution corresponds to two different "populations of experiments", each of them preferentially ending with the choice of one of the two shelters. Furthermore, in this case with two identical shelters, the symmetry of the U-shape means that each shelter is randomly chosen from one experiment to another. The dynamics of this choice can be seen in Figs. 4 B.1 and B.2. It shows that the choice occurs very rapidly within the first minutes of the experiments. It also shows that this choice is very strong, since $75.5 \pm 3.36\%$ (mean±s.e., $n = 20$) of the population of robots is under the chosen shelter at the end of the experiments ($78 \pm 0.53\%$, $n = 1000$, in simulations). Thus this set of experiments clearly shows that the aggregation process described above (with very simple individual behaviors) can lead a group of robots to perform a collective choice between two aggregation sites.

The two other sets of experiments were designed to assess the impact of a qualitative difference between the two shelters on the collective choice. As in the previous set of experiments, a group of 10 robots faced a choice between two shelters. But this time, while one of the shelters kept the same size as in the previous experiment, the size of the other was altered.

In a first set of 20 experiments, we confronted a 14 cm diameter shelter (able to house the whole robot population) with a 10 cm diameter shelter (too small to house the whole population of robots). As can be seen in Figs. 4 A.1 and A.2, robots quickly and strongly choose the shelter able to house their whole population. Thus, at the end of the experiments, $68 \pm 3.29\%$ (mean±s.e., $n = 20$) of the population is under the 14 cm diameter shelter ($72.7 \pm 0.79\%$, $n = 1000$, in simulations). The choice distribution shows a strong shift towards the 14 cm diameter shelter (see Fig. 3 A.2). This shift is the result of more than the simple difference between the area of the two shelters. Indeed, a comparison between the experimental distribution and a trinomial distribution (Fig. 3 A.1) taking into account this difference in size shows a strong difference ($\chi^2 = 365.4$,

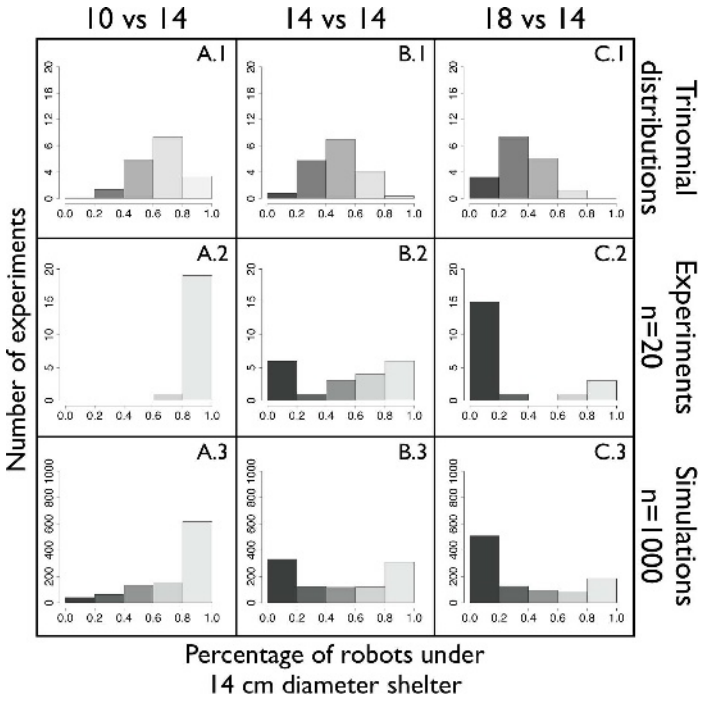


Fig. 3. Choice distributions. In these distributions, each block represents a number of experiments ending with a given percentage (0-20, 20-40, 40-60, 60-80 and 80-100 percent) of robots under one of the two shelters. **Top**: trinomial distributions (random choice). **Middle**: experimental distributions ($n = 20$). **Bottom**: simulation distributions ($n = 1000$). **Columns A and C** represent choice distributions for the 14 cm diameter shelter against either the 10 cm diameter shelter (column A) or the 18 cm diameter shelter (column C). For each of these distributions, blocks on the right mean choice of the 14 cm diameter shelter and blocks on the left mean choice of the other shelter (either 10 or 18 cm diameter). **Column B** represents the choice distribution for a 14 cm diameter shelter against an other 14 cm shelter.

$df = 4$, $p < 0.0001$). Similar results are obtained with simulations (see Fig. 3 A.3) and a chi square analysis of contingency tables shows no difference between experiments and simulations ($\chi^2 = 9.4$, $p = 0.0595$, p-value simulated with 10000 replicates [13]). The disappearance of the U-shape of the distribution means that it remains only one "population of experiments" preferentially ending with the choice of the 14 cm diameter shelter, i.e. the one able to house the whole population of robots.

In a second set of 20 experiments, we confronted a 14 cm diameter shelter with a 18 cm diameter shelter. Both shelters are able to house the whole population of robots. As can be seen in Figs. 4 C.1 and C.2, robots choose the 18 cm diameter shelter. Thus, at the end of the experiments, $70.5 \pm 7.56\%$ (mean±s.e., $n = 20$) of the population is under the 18 cm diameter shelter ($61 \pm 1.12\%$,

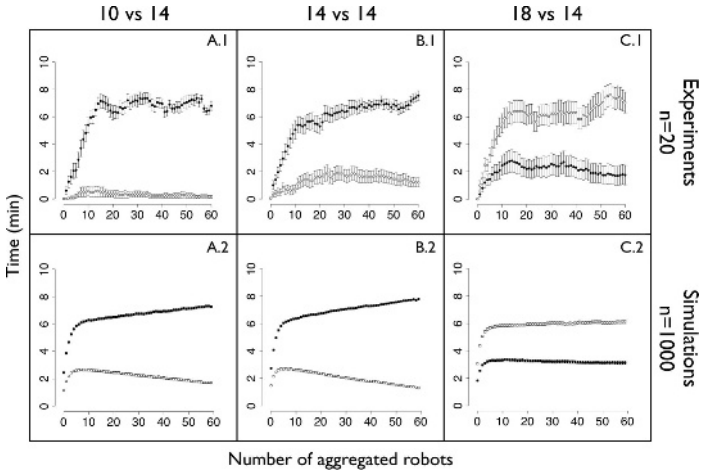


Fig. 4. Choice dynamics: number of robots aggregated under each shelter. **Top**: experimental data ($n = 20$). **Bottom**: simulation data ($n = 1000$). **In column A and C**, black dots represent data for the 14 cm diameter shelter; white dots represents data for either the 10 cm diameter shelter (column A) or the 18 cm diameter shelter (column C). **In column B**, black dots represent data for the chosen shelter (i.e. the shelter which is chosen at the end of each experiment); white dots represent data for the "not chosen" shelter. In all cases, each dot represents the mean ± s.e.

$n = 1000$, in simulations). The choice distribution shows a shift towards the 18 cm diameter shelter (see Fig. 3 C.2). This shift is the result of more than the simple difference between the area of the two shelters. Here also the comparison between the experimental distribution and a trinomial distribution (Fig. 3 C.1) taking into account this difference in size shows a strong difference ($\chi^2 = 373.8$, $df = 4$, $p < 0.0001$). And similar results are obtained with simulations (see Fig. 3 C.3) and a chi square analysis of contingency tables shows no difference between experiments and simulations ($\chi^2 = 5.4$, $p = 0.2301$, p-value simulated with 10000 replicates). But contrary to the previous experiment, the U-shape of the distribution has not disappeared and the two "populations of experiments" still exist: one that preferentially ended by a choice of the 14 cm diameter shelter, the other that preferentially ended by a choice of the 18 cm diameter shelter, the latter prevailing on the former.

From the two latter sets of experiments, we can conclude that the group of robots will choose preferentially a shelter that is sufficiently large to house all its members. But when the group is confronted with two sufficiently large shelters, the self-enhanced aggregation mechanism can lead the group to two stable choices, with a preference for the larger shelter. This implies that the group of robots is able to "sense" and "compare" the size of the shelters during the collective decision process, a performance that is beyond the direct scope of the simple aggregation process used in these experiments and that is not explicitly implemented in individual robots. We hypothesise that this relies on the higher

probability for the robots to encounter this shelter in the arena. Indeed, the more robots encounter a shelter, the more likely they will stop spontaneously under it. Thus, there will be more individual stopped robots under the bigger shelter that will act as "seeds" for new clusters.

4 Conclusion

In this work, we achieved the implementation of a biological model of self-enhanced aggregation in a group of mini-robots Alice. Despite the strong differences in terms of sensory abilities between biological and artificial models, the aggregation dynamics observed in robots closely match those observed in cockroaches. This result is obtained by measuring robot and cockroach behaviours in terms of behavioural probabilities, thus taking into account sensory and motor abilities of the two systems. Then, by calibrating the behavioural probabilities programmed in the robots, we reproduced both individual displacement and stop behaviours of the biological system with the artificial one. And the aggregation dynamics emerge from these individual behaviours, as is expected from the model described in [9,10]. With this method, it is thus only required that the robot features approximatively reproduce cockroach features to accurately reproduce their aggregation behaviour.

Moreover, we achieved a collective decision process from this simple biological model of aggregation. We showed that a self-enhanced aggregation process associated with a preference for a given type of environmental heterogeneity (here a preference for dark places) can lead a group of robots to a collective choice for an aggregation site. Furthermore, this choice can be related to a collective ability to "sense" and "compare " the sizes of the aggregation sites. This is a very interesting robotics example of an interaction between a simple self-organized mechanism and an evironmental template, leading to the emergence of a far more complex collective behaviour and of new collective abilities not explicitly coded in the basic model of aggregation.

This work opens some interesting perspectives for collective robotics. Collective choices could be associated, for instance, with an ordering behavior of the same kind than the one described in [8], so that robots would assemble objects of different types in different places. We argue that such associations are new challenges to take up if this collective robotics, based on self-organized mechanisms and/or biologically inspired behaviors, must become an efficient and robust way to achieve complex tasks with groups of numerous small autonomous robots.

References

1. W. Agassounon and A. Martinoli. A macroscopic model of an aggregation experiment using embodied agents in groups of time-varying sizes. In *Proceedings of the 2002 IEEE Systems, Man and Cybernetics Conference*, Hammamet, Tunisia, 2002. IEEE Press.
2. J.-M. Ame, C. Rivault, and J.-L. Deneubourg. Cockroach aggregation based on strain odour recognition. *Animal Behaviour*, 68(4):793–801, 2004.

3. E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence : from natural to artificial systems.* Oxford University Press, Oxford, 1999.
4. S. Camazine, J.L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. *Self-organization in biological systems.* Princeton University Press, Princeton, 2001.
5. G. Caprari, T. Estier, and R. Siegwart. Fascination of down scaling – Alice the sugar cube robot. *Journal of Micromechatronics*, 1(3):177–189, 2002.
6. J. L. Deneubourg, A. Lioni, and C. Detrain. Dynamics of aggregation and emergence of cooperation. *Biological Bulletin*, 202(3):262–7, 2002.
7. P.-P. Grassé. La reconstruction du nid et les coordinations inter-individuelles chez *Bellicositermes Natalensis et Cubitermes sp.* La théorie de la stigmergie : essai d'interprétation du comportement des termites constructeurs. *Insectes sociaux*, 6:41–81, 1959.
8. O. Holland and C. Melhuish. Stigmergy, self-organisation, and sorting in collective robotics. *Artificial Life*, 5:173–202, 1999.
9. R. Jeanson, S. Blanco, R. Fournier, J. L. Deneubourg, V. Fourcassié, and G. Theraulaz. A model of animal movements in a bounded space. *Journal of Theoretical Biology*, 225(4):443–451, 2003.
10. R. Jeanson, C. Rivault, J.-L. Deneubourg, S. Blanco, R. Fournier, C. Jost, and G. Theraulaz. Self-organized aggregation in cockroaches. *Animal Behaviour*, 69(1):169–180, 2005.
11. A. Ledoux. Étude experimentale du grégarisme et de l'interattraction sociale chez les Blattidés. *Annales des Sciences Naturelles Zoologie et Biologie Animale*, 7:76–103, 1945.
12. A. Martinoli and F. Mondada. Collective and cooperative group behaviours: biologically inspired experiments in robotics. In O. Khatib and J. K. Salisbury, editors, *Proceedings of the Fourth International Symposium on Experimental Robotics*, pages 3–10, Stanford, June 1995. LNCIS.
13. R Development Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria, 2004. ISBN 3-900051-07-0.
14. M. K. Rust, J. M. Owens, and D. A. Reierson. *Understanding and controlling the german cockroach.* Oxford University Press, Oxford, 1995.
15. E. Sahin. Swarm robotics: From sources of inspiration to domains of application. *LNCS*, 3342:10–20, 2005.
16. T.D. Seeley, S. Camazine, and J. Sneyd. Collective decision-making in honey bees: how colonies choose among nectar sources. *Behavioural Ecology and Sociobiology*, 28:277–290, 1991.
17. K. Sugawara and M. Sano. Cooperative acceleration of task performance: foraging behavior of interacting multi-robots system. *Physica D: Nonlinear Phenomena*, 100(3/4):343–354, 1997.
18. I. A. Wagner and A. M. Bruckstein. Ant robotics. *Annals of Mathematics and Artificial Intelligence*, 31:1–238, 2001.

(Co)Evolution of (De)Centralized Neural Control for a Gravitationally Driven Machine

Steffen Wischmann, Martin Hülse, and Frank Pasemann

Fraunhofer Institute for Autonomous Intelligent Systems,
Schloss Birlinghoven, D-53754 Sankt Augustin, Germany
{steffen.wischmann, martin.huelse, frank.pasemann}@ais.fraunhofer.de,
http://www.ais.fraunhofer.de/INDY

Abstract. Using decentralized control structures for robot control can offer a lot of advantages, such as less complexity, better fault tolerance and more flexibility. In this paper the evolution of recurrent artificial neural networks as centralized and decentralized control architectures will be demonstrated. Both designs will be analyzed concerning their structure-function relations and robustness against lesion experiments. As an application, a gravitationally driven robotic system will be introduced. Its task can be allocated to a cooperative behavior of five subsystems. A co-evolutionary strategy for generating five autonomous agents in parallel will be described.

1 Introduction

The *Artificial Life* (AL) approach to *Evolutionary Robotics* (ER) provides promising methods for optimizing a variety of control problems [8,11]. This includes the optimization of structure and parameters of artificial recurrent neural networks (RNNs), morphology parameters of robots, or even co-evolution of many different populations. Within our approach to AL and ER we are using evolutionary techniques for generating RNNs controlling robot behavior [9]. We are aiming at artificial systems with so called *minimal cognition* [2,5]. In this context we are trying to deal with minimal models of non-linear dynamical control that can offer a variety of behavioral patterns. To investigate the dynamical properties of such control structures we study relatively simple artificial systems to gain deeper insights into the essence of dynamical systems such as RNNs and robot-environment interactions.

In this paper we will present an example offering investigations of basic cooperation mechanisms among artificial agents coupled through a common body. There are many examples, where cooperation within a group of homogeneous or heterogeneous agents may have advantages over single agents in solving complex tasks [3]. One reason is the possibility of task decomposition and task allocation. To give an example for task allocation, we use the artbot *micro.eve* (http://www.sphericalrobots.com). Among other intentions, Julius Popp designed *micro.eve* to provide a benchmark system for control architectures

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 179–188, 2005.

where a *simple body-consciousness emerges*. Considering *embodiment* as a fundamental aspect of creating artificial autonomous agents [4,10], the system can be described as a set of five agents, which are able to act locally and independently. Hence the agents are connected to a common body, their local actions affect not only themselves but also the behavior of the common body and consequently also the behavior of the other agents. They have to cooperate to solve the task, which is a well known problem in collective robotics [7,12]. For instance, in [1] an example of physically linked robots solving a common task is given. Here, the system seems much simpler. But due to this simplicity it allows detailed analysis of the underlying control structure. We will show two different kinds of control. First, we evolve one central RNN which controls all the agents. Second, we evolve a RNN for each separate agent in co-existing populations, whereas each agent can sense the action of the other agents only by sensors providing information about the common bodies' behavior. There is no explicit communication among the agents. We will analyze two resulting architectures with respect to performance, structure-function relations, and robustness.

2 Methods

Figure 1 illustrates the artbot *micro.eve* as well as its simulated model. The robot consists of five movable arms, which are connected to a common body. The center of mass of these arms can be actively translated by a servo motor. Through a coordinated motion of the five arms the overall center of mass of the robot can be translated in such a way, that the ring starts to rotate on two supporting rollers. Here, we defined the task of the system as rotating as fast and as harmonically as possible.

The control structures have to produce the motor signal for each arm. The sensory system consists of potentiometers in each motor, a gyroscope located inside the ring, and five hall sensors equally distributed over the ring. Each hall sensor is located between two arms, respectively. These hall sensors are binary

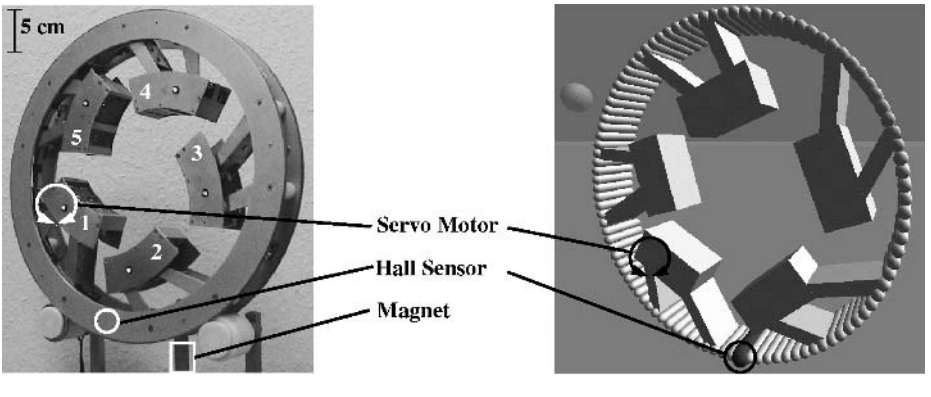


Fig. 1. The artbot *micro.eve*. Left: Hardware, Right: Simulation

switches emitting a peak if they pass a magnet placed at the bottom of the ring. We merged all five hall sensor to one sensory input relative to the arm index, i.e. for each arm the next hall sensor to the right has the index 1. The arm and hall sensor indices are incremented counterclockwise. The mapping (i, s_i), where i denotes the index of the last activated hall sensor and s_i the according sensor value, is as follows: $(0, 0.0); (1, 1.0); (2, 0.66); (3, 0.33); (4, -1.0); (5, -0.33)$. Hence the output of the input neuron, which provides the hall sensor information, should be zero if no hall sensor is activated at all ($i = 0$) the use of this discrete mapping was chosen instead of a monotonic function. Accordingly, within the decentralized control approach each agent has it its own sight of the hall sensor. To reduce the amount of input neurons for the centralized control structures the sight of the first arm is provided as sensory input. The sensor values of the potentiometers are linearly mapped to the interval $[-1.0, 1.0]$. The gyroscope values are transformed to an angular velocity of the ring with a maximum at 0.5 rounds/secs. For the input of the neural network these values are also mapped to the interval $[-1.0, 1.0]$, where negative values indicate a counterclockwise and positive values a clockwise rotation.

For our control approach we used discrete-time artificial recurrent neural networks with standard additive neurons with sigmoidal transfer function $\sigma = tanh$. For generating these controllers we used an evolutionary algorithm that allows variation of the network's structure and its parameters at the same time. A full description of the algorithm and some other applications can be found elsewhere [6]. Here, to solve the robot's task we defined the following fitness function:

$$F = |\overline{\omega}| * (1 - \frac{\sum_{t=0}^{n} |\omega(t) - \overline{\omega}|}{2n}) \tag{1}$$

where ω is the angular velocity of the ring represented by the output of the mapped gyroscope sensor value. The left term ($|\overline{\omega}|$) rewards a high mean velocity and the right term rewards a harmonic rotation. Due to the use of $tanh$ as transfer function the output of the velocity sensor neuron is in a range between -1 and $+1$. Accordingly the range of the fitness value is between 0.0 and $+1.0$. Note that we are using an implicit fitness function, i.e. no global knowledge is used. Parameters of the fitness function are solely determined by sensor information that are accessible by the agent.

To avoid dominance of specialists we evaluated every individual on 20 different randomly (uniformly distributed) generated starting conditions. For this purpose the ring angle is varied in the full range of 2π, and the angle of each arm is varied within its complete working range. The resulting fitness value is the mean fitness of these 20 evaluation cycles. One evaluation cycle lasted 1200 evaluation steps (corresponds to 120 secs real time).

For generating the decentralized control architectures we applied a simple co-evolutionary strategy. Each arm is controlled by one autonomous RNN. The sensory input of the RNNs is reduced to one relative hall sensor input, as previously described, the ring velocity, and the arm's potentiometer. A single controller has one output neuron controlling the according motor signal. Every agent

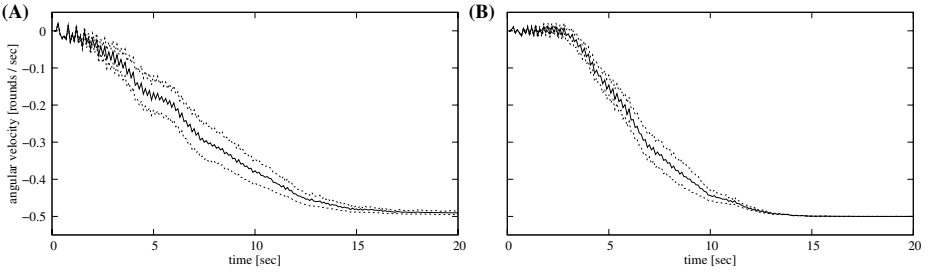


Fig. 2. Evolution of the angular ring velocity with time and their variance for the decentralized (A) and centralized control structures (B), starting from several initial conditions.

is evolved in a separate population. The evolutionary process for a single population is the same as for the evolution of the centralized control structure, i.e. every population has its own selection, reproduction, and variation operators. In every population the individuals are sorted according to their fitness values, starting with the highest. The offsprings get no fitness value after selection, they are appended at the end of the sorted list. For evaluation, one agent of each population is selected and applied to the arm related to its population, i.e. a group of five agents, each from a different population, is evaluated together at the same time. The selection of the group members is rank based related to the fitness value, i.e. the agents taking the first place in each population are evaluated together, than the agents on the second place and so on. In such a way the evaluation of i populations needs j evaluation cycles, where j is the number of individuals within the largest population. The fitness function (equation (1)) regards the performance of a group of agents, which have to cooperate. Therefore every agent within one group gets the same fitness value, regardless which local acting network gives the most or even the least contribution to it.

3 Results and Discussion

Figure 3 shows the outcome of the evolutionary process for the centralized and decentralized control approach. Both control techniques are successful in solving the given task. The mean fitness $\overline{F}$ and its variance σ^2 for 100 evaluation cycles with random starting conditions is 0.900 ($\sigma^2 = 0.001$) for the centralized and 0.884 ($\sigma^2 = 0.003$) for the decentralized control architecture. After transferring these RNNs unchanged to the real machine, the observed behavior was qualitatively the same, although the evolution of these RNNs was completely done in simulation with a very simplified motor model, and only roughly approximated friction and noise. We observe that once the ring started to rotate from a given starting position, it harmonically rotates for the whole evaluation time, i.e. a mean fitness of 0.9 indicates that this agent can handle about 90 percent of different starting conditions. Note, that it is nearly impossible to reach a fitness value of 1.0 due to the time the agent needs to initiate a rotation. During this

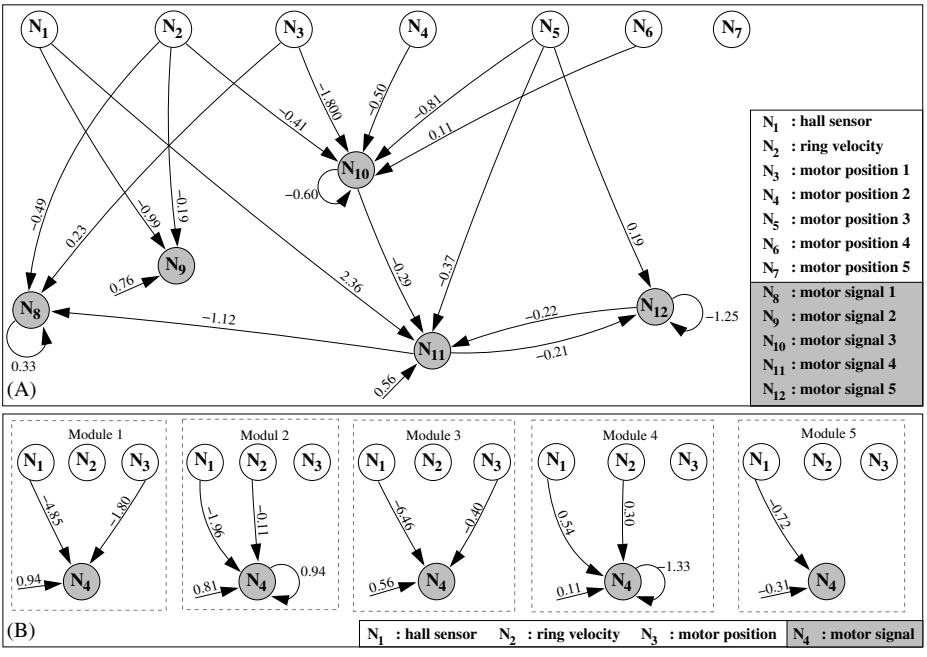


Fig. 3. RNNs of the centralized (A) and decentralized (B) control architectures. Input neurons are indicated by white circles and output neurons by grey circles.

initialization the rotation cannot be harmonic. Therefore the first 10 seconds of an evaluation cycle did not contribute to the fitness value.

Figure 2 illustrates the evolution of the angular ring velocity with time. The average of 100 runs with random, uniformly distributed, starting conditions (ring angle, arm angles) and the variance are given for both control architectures. The centralized RNN as well as the decentralized RNN performs only counterclockwise rotation and approaches the maximum angular velocity ($|\omega_{max}| = 0.5 rounds/sec$) in about 10-15 seconds. The decentralized RNN seems to have a smoother transition to ω_{max} whereas the centralized RNN needs about 2.5 seconds for initializing a rotation at all but than reaches ω_{max} slightly faster.

In the following we will discuss the dynamics of the RNNs and their effect on the robot behavior. Focusing on the motor neurons, we can determine two main mechanisms controlling two behavioral states: (1) Oscillations to initialize the rotation, and (2) Strong impact of the hall sensor to maintain the rotation.

Looking at figure 3 and 4, we find one neuron with a period-2 oscillation in both control structures. For the centralized control this is N_{12}, and for the decentralized control N_4 of module 4. The period-2 oscillation is due to a over-critical negative self-connection and persists for all the time (see figure 4). The strong impact of the hall sensor within the other modules of the decentralized control can be seen in figure 3. For the centralized RNN N_9 and N_{11} are mainly controlled by the hall sensor input, N_8 receives its strongest input from N_{11}, and

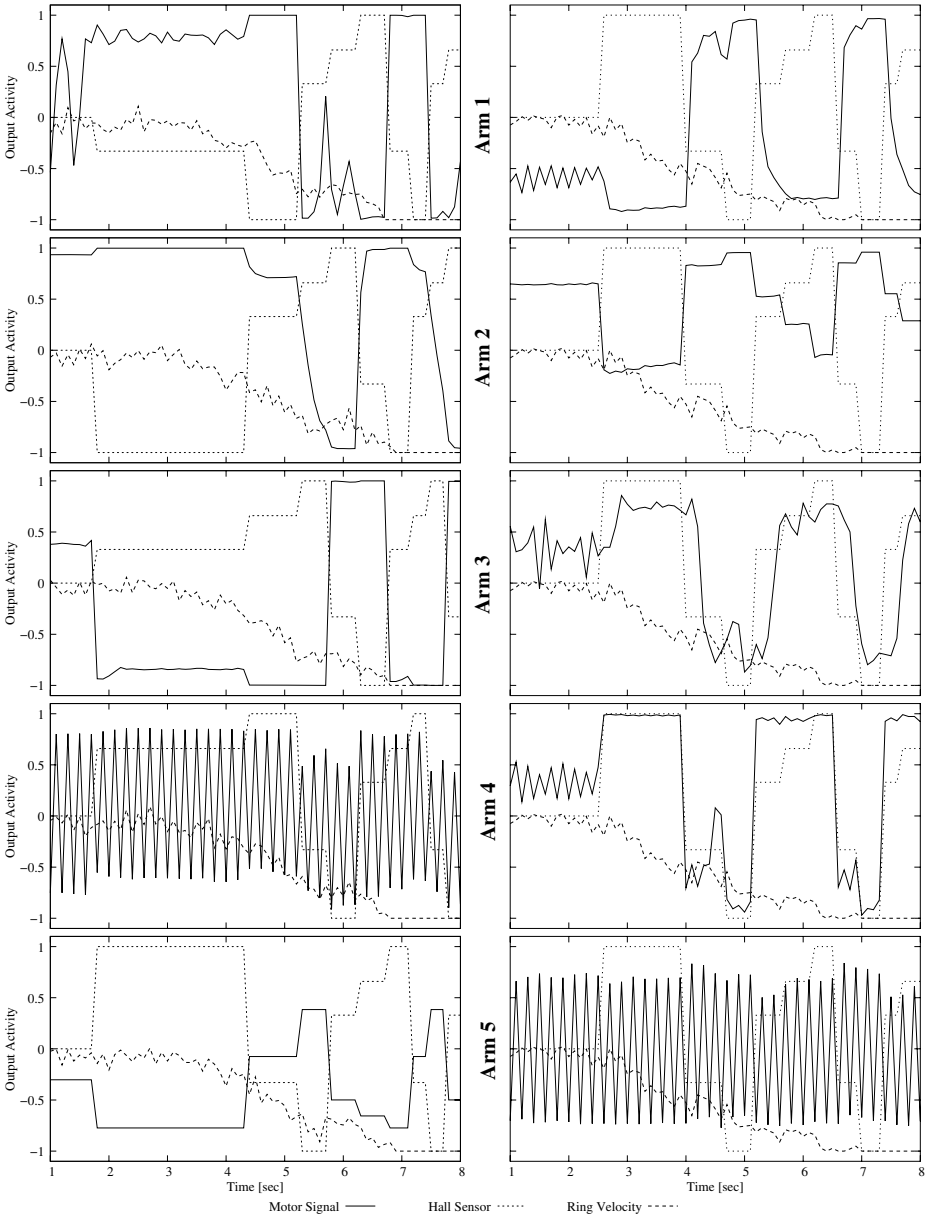


Fig. 4. Motor signals, hall sensor and ring velocity input of the decentralized (left) and centralized (right) control structures. The according neurons can be found in figure 3.

N_{10} gets a strong input from N_3 which is directly influenced by the output of N_8. In figure 4 we can see the correlation between the hall sensor input and the signal of all these motor neurons.

What does this mean for the behavior of the robot? At the beginning the hall sensor is inactive, hence the signal of the according input neuron is zero. Referring to figure 4, most of the motor neuron signals stay around an output value according to their bias terms or the input of other neurons. Therefore the overall center of mass of the ring is translated once and thus, the ring may rotate a little. Depending on the starting condition this rotation may be enough to activate a hall sensor, if this is near the magnet (see figure 1). If the hall sensor stays inactive, the rotation will stop because most of the motor signals depends on this input and the according arms will not move at all. In this case the period-2 oscillating motor neurons will move the ring very slowly until the hall sensor is activated. Than the rotation starts and is maintained as it can be seen from figure 4. For most of the time we observe two pairs of output neurons producing opposed signals. This means that two arms translating their masses to the center of the ring, and at the same time two other arms translate their masses to the ring periphery. This action maintains the ring rotation. In the decentralized control modules the two pairs get negative feedback from their hall sensory input, in contrast to the centralized control, where we find positive and negative feedback leading to the opposed movements. In the decentralized control, the opposed movement is based on the fact that each module has its own, relative, sight of the hall sensor as described in section 2, whereas the centralized control has only one sight of the hall sensor.

What is the advantage that one neuron is oscillating all the time? We can find a good reason, if we perturb the system, i.e. manually stop the ring rotation. Hence, most of the motor neurons are mainly influenced by the hall sensory input, as we saw before, most of the arms will stay at their positions when the ring stops because the hall sensor stays active at its last value. If we than release the ring, a slow movement can be observed, due to the still oscillating motor neuron, until the hall sensor changes its value, and that gives rise to a change in the other motor signals until the ring rotates harmonically again.

There is another interesting fact in the presented control structures. We saw that there are not as many interesting dynamical features as one could expect by using RNNs, which is possibly due to the simplicity of the task. On the one hand we have a period-2 oscillator and on the other hand the behavior of the others neurons seems to depend more or less directly on one sensor input. But for instance, if we look at the motor neuron of module 1 in the decentralized control architecture, we see, while the hall sensor is still inactive, the occurrence of an oscillation (see figure 4) which is not provided by the structure of the RNN. This oscillation is based on the input neuron which provides information about the position of the arm, which is controlled by the motor neuron. At the beginning the arm moves to a certain position according to the bias term, but than the motor neuron gets a strong negative feedback from the position sensor leading to an opposed movement and so on. Here we have a negative *feedback loop through the environment* that is depressed when the hall sensor becomes active due to the much stronger connection from this input neuron.

Table 1. Mean performance and its variance of the lesion experiments

Lesion	Centralized Control		Decentralized Control	
	$\overline{F}$	σ^2	$\overline{F}$	σ^2
none	0.900	0.001	0.884	0.003
Motor 1	0.005	0.000	0.740	0.121
Motor 2	0.702	0.120	0.776	0.018
Motor 3	0.739	0.060	0.671	0.085
Motor 4	0.006	0.000	0.572	0.185
Motor 5	0.579	0.140	0.753	0.082

It is interesting that we can detect similar control principles in two different designs of control. But then one can ask, what is the advantage of one method over the other. First, we can see decentralization leads to less complexity in the structure of the single RNNs (figure 3). And second, due to the autonomy of the single control modules, there is no explicit communication between the decentralized control units, as we discussed it for the centralized RNN, which may lead to more robustness. To demonstrate the second point lesion experiments were done with the two introduced results of the different control designs. Both architectures were tested on 100 random, uniformly distributed, starting conditions. Each run lasted 1200 evaluation steps, i.e. 2 minutes in real time. Here, lesion means the fixation of one motor neurons output to zero value during the whole evaluation time. This could be considered as the simulation of a motor breakdown of the real hardware. Table 1 gives the mean performance, calculated with the fitness function (equation 1), and its variance for all lesion experiments performed on both control architectures.

Considering the centralized control structure, lesions of motor 1 and 4 have the strongest impact. The agent can handle no starting condition. If we look at the structure (figure 3 (A)) of the RNN, we can see that setting the output of N_8 to zero leads to zero output of N_3 as well, which consequently has a strong impact on N_{10}. Setting the output of N_{11} to zero will also have a strong influence on N_8 due to the strong connection between these two neurons. We can see, the fixation of one output neuron also affects the dynamics of other output neurons. On the contrary, the decentralized control structures have no inter-connections between the output neurons, which is reflected in the results of the lesion experiments. In the worst case it still can handle about 57 % of the starting conditions. The worst case is the lesion at the oscillating neuron (module 4). Interestingly, lesion of the oscillating output neuron N_{12} of the centralized control structure leads to a similar performance. That the system is still able to handle about half of the different starting conditions in this case is due to the initialization process described earlier in this section. For lesion of motor 2 and 3 at the centralized control structure, where the according motor neurons have no or only a small influence on other neurons, we observe a performance comparable to the decentralized one.

It is important to note that the described perturbations were not part of the boundary conditions during the evolutionary process. If these perturbations were

included, we would expect a more robust behavior of the centralized control as well. But one should be aware of that it could be highly difficult to pre-estimate every possible kind of perturbations in real systems. In the presented example the robustness against motor breakdowns is an innate property of the decentralized control approach.

4 Conclusions

In this paper we demonstrated the evolution of a centralized and decentralized control architecture for the gravitationally driven artbot *micro.eve*. Both were able to successfully solve the given task. We could show that minimal structures arise out of the evolutionary process without any prior assumptions about the structure of the RNNs. By analyzing the structure-function relations, we identified similar control principles in both designs, mainly based on two mechanisms, periodic oscillations, mainly responsible for a robust initialization and robustness against perturbations, and a strong coupling of the hall sensory input, mainly responsible for maintaining the ring movement. We found that this strong coupling is also determined by inter-connections between the output neurons in the centralized RNN. Because for the decentralized control no communication was allowed, this mechanism was determined by direct connections to this sensor input. Due to this fact we could show that the autonomy of the subsystems of the decentralized control approach leads to more robustness against motor breakdowns. We saw that the action of one agent indirectly influences the action of the other agents. We think, it should be possible to handcraft a centralized control structure containing similar autonomous subsystems. But we argue that it is hard to expect such a result from an evolutionary process if no specific assumptions about the structure and parameters of RNNs are made. By identifying the main properties of the control structure one could manually transfer the properties of decentralization, such as autonomous substructures, to centralized control architectures. Another step would be to evolve homogeneous decentralized control structures, where it should be more obvious that their properties are applicable to a centralized control exhibiting more robustness to lesion experiments.

We saw that decentralization gives rise to less complexity, concerning the structure of the RNNs, and more fault tolerance in the presented example. We are not claiming that these principles could be generalized for all kinds of decentralized control problems. But as it is known, these issues are crucial points for many of other examples concerning decentralized control as well [13].

Even though the presented results do not directly lead to general solutions of problems in AL and ER, they provide a simple model of *minimal cognition* for an unconventional machine. Due to the few system parameters it provides a platform for studying first steps in neural control of autonomous robots, basic cooperation mechanisms, and robot-environment interactions. Furthermore, we showed how to apply the same evolutionary algorithm to the development of centralized and decentralized control architectures. We introduced a simple co-evolutionary strategy, for which the evaluation time does not increase with the

number of populations. We are aware of the fact that evolution of competitive behavior may require a more complex evaluation strategy, but to solve simple cooperative tasks the presented strategy is sufficiently good.

References

1. G. Baldassarre, D. Parisi, and S. Nolfi. Coordination and behavior integration in cooperating simulated robots. *Proc. of the 8th International Conference on the Simulation of Adaptive Behavior (SAB-2004)*, pages 385–394, 2004.
2. R. D. Beer. The dynamics of active categorical perception in an evolved model agent. *Adaptive Behavior*, 11(4):209–243, 2003.
3. Y. U. Cao, A. S. Fukunaga, and A. B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.
4. A. Clark. *Being There: Putting Brain, Body, and World Together Again*. MIT Press, Cambridge, MA, USA, 1996.
5. I. Harvey, E. A. Di Paolo, R. Wood, M. Quinn, and E. Tuci. Evolutionary robotics: a new scientific tool for studying cognition. *Artificial Life*, 11:79–98, 2005.
6. M. Hülse, S. Wischmann, and F. Pasemann. Structure and function of evolved neuro-controllers for autonomous robots. *Connection Science*, 16(4):249–266, 2004.
7. A. J. Ijspeert, A. Martinoli, A. Billard, and L. M. Gambardella. Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment. *Autonomous Robots*, 11(2):149–171, 2001.
8. S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, Cambridge, USA, 2000.
9. F. Pasemann, U. Steinmetz, M. Hülse, and B. Lara. Robot control and the evolution of modular neurodynamics. *Theory in Biosciences*, 120:311–326, 2001.
10. R. Pfeifer. On the role of embodiment in the emergence of cognition: Grey walter's turtles and beyond. *Proc. of the workshop The legacy of Grey Walter, Bristol, 2002.*, 2002.
11. R. Pfeifer and C. Scheier. *Understanding Intelligence*. MIT Press, Cambridge, USA, 1999.
12. M. Quinn, L. Smith, G. Mayley, and P. Husbands. Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors. *Philosophical Transactions of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 361:2321–2344, 2003.
13. M. Resnick. *Turtles, termites, and traffic jams: explorations in massively parallel microworlds*. MIT Press, Cambridge, USA, 1994.

Co-evolution of Structures and Controllers for Neubot Underwater Modular Robots

Barthélémy von Haller[1], Auke Ijspeert[1], and Dario Floreano[2]

[1] Biologically Inspired Robotics Group
[2] Laboratory of Intelligent Systems,
Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland
{barthelemy.vonhaller, auke.ijspeert, dario.floreano}@epfl.ch
http://birg.epfl.ch, http://lis.epfl.ch

Abstract. This article presents the first results of a project in underwater modular robotics, called Neubots. The goals of the projects are to explore, following Von Neumann's ideas, potential mechanisms underlying self-organization and self-replication. We briefly explain the design features of the module units. We then present simulation results of the artificial co-evolution of body structures and neural controllers for locomotion. The neural controllers are inspired from the central pattern generators underlying locomotion in vertebrate animals. They are composed of multiple neural oscillators which are connected together by a specific type of coupling called synaptic spreading. The co-evolution of body and controller leads to interesting robots capable of efficient swimming. Interesting features of the neural controllers include the possibility to modulate the speed of locomotion by varying simple input signals, the robustness against perturbations, and the distributed nature of the controllers which makes them well suited for modular robotics.

1 Introduction

This article presents an adaptive scheme for underwater navigation of modular robots based on the artificial co-evolution of body structures and neural controllers for locomotion. The modular robots used in this paper are part of a long-term project [1] aimed at conceiving robots capable of self-construction and self-reproduction, as first proposed by John von Neumann. Therefore, we call such robots *Neubots* from von NEUmann roBOTS, which also means New Robots in German.

The neural controllers studied in this paper are central pattern generators (CPGs), which are networks capable of producing coordinated patterns of rhythmic activity while being modulated by simple non-oscillatory signals. Our CPGs represent an interesting framework for modular robotics because of (1) their distributed nature, (2) the locality of their interactions, (3) their robustness against perturbations, and (4) their ability of coordinating multiple degrees of freedom while being modulated by simple input signals.

In this article, the controllers are designed incrementally, with first the design of a neural oscillator which serves as the building block for the complete CPGs.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 189–199, 2005.

In a second stage complete CPGs are co-evolved with the body structure. During this second stage, the neural oscillators embedded into each robot unit are coupled to oscillators in neighboring units using synaptic spreading, which corresponds to projecting connections between two types of neurons to the same type of neurons in neighbor neural oscillators. This type of intercoupling is well-suited for modular robotics because of its locality and because it can be described in fewer parameters than an all-to-all coupling between oscillators.

Neubots and Related Modular Robots. More than 50 years ago, John von Neumann investigated the possibility of designing physical robots that can self-assemble and self-reproduce. He arrived to the conclusion that such robots should be composed of a dozen different types of simple modules produced in hundreds of thousands of copies [3]. Von Neumann also argued that the control system of such modules should be composed of some sort of McCulloch-Pitts neurons [4]. However, von Neumann abandoned this line of research because of technological limitations of that time and concentrated on the computational aspects of such systems, which eventually resulted on the birth of cellular automata. We think that today's technology and science of complex self-organizing systems is ripe for the realization of physical self-assembling and self-reproducing robots. In the Neubots project, which is described more extensively elsewhere [1], we continue from where von Neumann left and redefine some of his intuitions in light of recent scientific and technological advancements.

The Neubot project rests on three main principles: 1) an heterogeneous and large pool of simple and specialized modules that can be combined in various ways to form a multicellular artificial organism; 2) a set of mechatronic and control mechanisms that allows the active recruitment and release of modules by the growing and self-reproducing organism; 3) a process of intrinsic and open-ended evolution of the organism mediated by its cells (modules), which possess the entire genome of the organism. In order to simplify the recruitment and release of modules, we conceived the early prototypes as underwater units. In this paper we investigate candidate solutions for simple navigation of the Neubot modules and we focus our investigation only on one specific module, which operates as a joint. The modules are made of faceted hull with 6 actuators (Figure 1). The connectors are based on a magnetic system composed of one permanent magnet and three yokes. A motor allows switching connectors between attractive, repulsive, and neutral states. Physical prototypes of the module have been constructed and tested [2], but only simulation experiments will be presented in this article.

Neubots belong to the larger family of self-organizing modular robots. In particular, the Neubot modules described here are similar to the M-Tran and Hydron robotic systems. The M-Tran system is composed of several identical modules that can connect to each other by means of active magnetic surface [6]. Robots made of such components can autonomously transform themselves into different shapes and use the joints to walk. However, the detaching process requires a lot of energy and is rather slow. The Hydron robotic system instead is composed of several identical waterproof and spherical units that are suspended

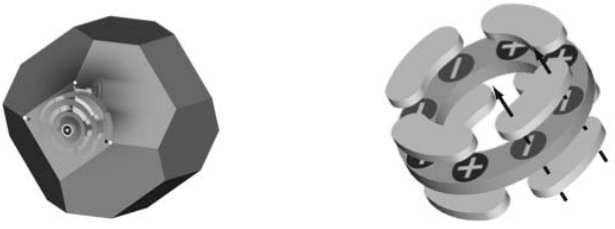


Fig. 1. Left: Schematic view of a Neubot's module and its actuation. **Right**: Schematic view of the connector.

in water and can move by means of water jets. These robots are currently used for exploring principles of self-organization [7], but cannot connect to each other. The Neubot modules used here are underwater units that can connect to each other by means of active magnets that require considerably less energy to detach and attach. Furthermore, the contact interface of the Neubot modules can rotate, thus permitting a complex articulation of the entire system.

Adaptive Locomotion for Modular Robots. A promising way to control locomotion is provided by the *Central Pattern Generators* (CPGs) observed in invertebrate and vertebrate animals [10].

CPGs have been used in the modular M-Tran system described above [11,12], in "monolithic" robots [13], and in simulated robots [14]. CPGs are interesting for modular robotics because of their distributed nature and their ability to generate efficient locomotion for complex multi-DOF structures while being modulated by simple control signals. One of the novelties of our approach is the use of neural oscillators connected by synaptic spreading (see next sections). The approach is well suited for a distributed implementation and for the optimization by a genetic algorithm.

In this paper we will co-evolve the body structures and the locomotion controllers of the modular robots in a three-dimensional simulation. Other examples of co-evolution include Karl Sims seminal work [8], Framsticks, a three-dimensional simulation project which offers various genotypes and fitness functions, to co-evolve morphology and control of virtual stick creatures [15], the work by Pollack and colleagues [16], and other interesting projects [17]. Our approach differs from previous work mainly in the type of neural controllers that we use (see above).

2 Co-evolving Structures and Controllers for Locomotion

2.1 Neuronal Simulation of the CPG, Evolution of Oscillators

The CPG models are designed incrementally. In a first stage, we evolve a canonical neural oscillator, which produces stable oscillations and whose frequency can be adapted by simple tonic signals. In a second stage, we co-evolve the body structure and the neural controller of multi-unit robots.

As in [14], neural oscillators are evolved from neural networks which have a left-right symmetry (cf Fig. 2) with three neurons of different "type" (A, B and C) and one motoneuron (M) on each side. Each neuron receives a tonic (i.e. non-oscillating) input BS representing the driving signals produced by the brain stem in vertebrates.

Neuron model. The neuron units are modeled as leaky integrators. According to this model, the mean membrane potential m_i and the short-term average firing frequency x_i of a neuron i are governed by the equations:

$$\tau_i \frac{\mathrm{d}m_i}{\mathrm{d}t} = -m_i + \sum_j w_{i,j} x_j \qquad \text{and} \qquad x_i = (1 + e^{-(m_i + b_i)})^{-1}$$

where b_i is the neuron's bias, τ_i is a time constant associated with the passive properties of the neuron's membrane, and $w_{i,j}$ is the synaptic weight of a connection from neuron j to neuron i. Both the neuron parameters and the synaptic weights are evolved.

Genetic Algorithm. GALib [20] was used for the real-number genetic algorithm (GA). The GA begins with a population of $N = 100$ randomly created individuals. At each generation, crossover, mutation and pruning operators are applied for creating $C = 50$ new children. For parent selection we used a rank-based roulette wheel method which chooses a parent with a probability which is inversely proportional to its rank. The crossover operator takes two parents and for each position exchanges the genes at that position with probability $P_C = 0.5$. Mutation changes, with a probability $P_M = 0.4$, the real value of a gene according to a Gaussian distribution (SD = 1.0) around the old value. The pruning operator is specific to the neural network optimization and prunes a connection (probability $P_P = 0.05$) by setting the gene corresponding to the weight of this connection to zero. The children are then evaluated and, as the population size is fixed, the $C = 50$ worst individuals are rejected from the total population (parents and children). The GA is stopped when the difference between the current best-of-generation score and the one 10 generations back in time is less than 1 percent.

Encoding. The parameters of a neural network are encoded into chromosomes which are fixed-length strings of real values. The genome encodes both the neural parameters of each neuron type —the time constant (τ), the bias (β), and the sign (excitatory (+) or inhibitory (−))— and the connectivity of the network — the synaptic weights of the outwards connections from itself to other neurons (including self-connections) and the synaptic weights from the tonic drive (BS, left and right). The motoneurons M are forced to be excitatory and do not have connections to other neurons. The total number of genes is 43.

Fitness Function. The fitness function is defined to reward solutions that (1) oscillate regularly with left and right motoneurons out-of-phase, (2) have as few connections as possible, and (3) have a frequency of oscillation that can be varied monotonically with the level of BS. Mathematically, the fitness function is a product of six factors:

$$fit = nb\_connect \cdot oscil \cdot regularity \cdot oscil\_phase \cdot freq\_range \cdot ampl$$

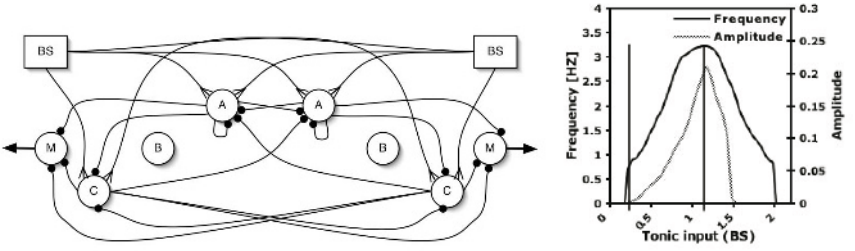


Fig. 2. Left: The neuronal oscillator. A black point indicates an inhibitory connection, a fork indicates an excitatory connection. The neurons of type B are not connected to any other neuron and can therefore be removed from the network. **Right**: Evolution of the frequency and amplitude when varying the input drive (BS). The vertical lines determine the region in which the amplitude and frequency increase monotonically.

where *nb_connect* corresponds to the inverse of the number of connections from a neuron, *oscil* to the number of optima (in order to favor oscillating networks), *regularity* to the inverse of the Standard Deviation of periods, *oscil_phase* to $|\theta_{oscil} - 0.5|$ with θ_{oscil} the phase between the left and right motoneuron, $freq\_range$ to max_freq/min_freq with $min\_freq \geq 0.8$ and finally *ampl* corresponds to the mean of amplitude. Each factor varies linearly between, and is bounded by, 0.05 and 1.0.

The network is evaluated during 6 seconds after a stabilization time of 6 seconds. The original input (BS) is set arbitrary to 1.0, but the network is evaluated multiple times with small BS increments and decrements in order to evaluate the capacity of the network to modify the amplitude and the frequency monotonically with the input.

Results. Fifteen runs were carried out until convergence (approximately 1000 generations per run) with populations of 100 randomly initialized individuals. All runs converged to networks capable of generating regular oscillations whose frequency can be modulated using the tonic drive (BS). One of them, the best, had the same topology as the network found in [14], but with small differences on the weights and thus on the neural activity.

Since the second design step is to evolve the connections between multiple neural oscillators by projecting internal connections to neighboring oscillators, it is important that the number of internal connections (from a neuron of type A, B or C to neurons of type A, B, C or M) is as small as possible. We therefore chose another one, with a smaller range of frequencies but with only 8 internal connections instead of 14. Figure 2 shows the topology of the chosen evolved network. The level of tonic drive (BS) can be varied between 0.2 and 1.13 (the frequency and the amplitude increase monotonically in this range of input), and a range of frequencies from 0.8 to 3.3 Hz can thus be covered. The amplitude then ranges from 0 to 0.21. Each neural oscillator will drive one motor in the mechanical structure by using the difference between left and right motoneurons

(M_L and M_R)): i.e. the desired angle (in radians) is defined as $\theta = \gamma(M_L - M_R) + \delta$ where δ is an angular offset and $\gamma = 5$.

2.2 Co-evolution of Body and Brain

In this section, the structure of simulated multi-unit robots and their locomotion controllers are co-evolved. The locomotion controllers are constructed out of the neural oscillators presented in the previous section, with one oscillator per robotic module. The coupling between the different neural oscillators is based on *synaptic spreading* similar to that used to model the lamprey's CPG in [5,9,14]. The idea is to project the connection between two neurons within one oscillator to corresponding neurons in neighboring oscillators. The synaptic spreading can be to the nearest neighbor oscillator only or even further.

An interesting property of this type of coupling is that it is specified by relatively few parameters compared to a scheme using all possible connections between different neural oscillators. Synaptic spreading only requires for each connection within one oscillator integers determining the extents of the projections (i.e. to first, second, third, ... neighbors).

Genetic Algorithm and Encoding. The genetic algorithm is the same as the one used for the neural oscillator but with several different parameters: $P_C = 0.2; P_M = 0.05; P_P = 0.0$, and a probability of structural mutation $P_{struct} = 0.025$ is added.

The genotype is a tree (no cycle allowed) as in [8] and [18], each node representing a module (see fig. 3 for details). It is thus a direct encoding which strongly correlates the phenotype and the genotypes. The simulation and evolution environment uses the genotype as the internal representation of the robot. In addition, a chromosome of real numbers, which we will call High Level Parameters (HLP), encodes the left and right input drive (BS). These genes specify the best BS in order to achieve the fastest locomotion. Crossover is simply done by exchanging two randomly chosen subtrees of the parents.

Fitness Function. The individuals are tested during 30 simulated seconds in a simulated 3D world. The simulation is a physics-based simulator (articulated rigid body dynamics with a simplified hydrodynamics model) built using ODE [19]. Throughout the evaluation period, the robot must cover as long a distance as possible. As the simple measure of the straight distance from the initial location to the final location seems not to be sufficient because of the risk to return to starting point after 30 seconds, the fitness function is based on the covered distance plus the cumulated distance:
$f = \alpha \cdot \|\boldsymbol{p(N)} - \boldsymbol{p(1)}\| + \beta \cdot \sum_{t=1}^{N-1} \|\boldsymbol{p(t+1)} - \boldsymbol{p(t)}\|$ where $p(t)$ is the t^{th} point sampled on the trajectory of the robot, N is the total number of recorded points, and $\alpha = 1$ and $\beta = 0.3$ are coefficients that modulate the weights of the absolute and integrated distances.

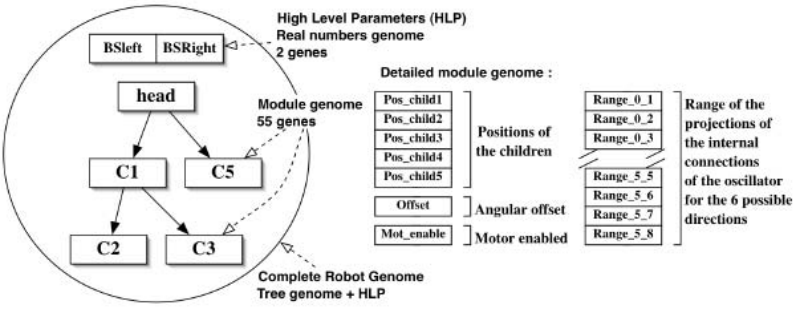


Fig. 3. Genome of a robot: It is designed as a tree with a node for each module of the robot. As we use a tree, each module has one and only one face which is actuated (the face attached to the parent) and therefore only one neural oscillator. Thus each node contains a chromosome of 55 genes containing the following information: positions of the children (5 binary genes whether or not a child module is attached to one of the 5 faces), angular offset of the joint between the module and its parent (1 real number gene), one binary gene indicating if the motor of the module must be actuated, and 48 integer genes encoding the synaptic spreading, i.e. coding the extent of the projections in the six possible directions for each oscillator.

2.3 Results

Two sets of 5 runs each, called A and B (view table 1), were carried out until convergence with a population of 100 robots. The first generations of the runs A and B start with randomly initialized populations of 5-unit robots, and 10-unit robots, respectively. Of course, the size of the robots can then be increased or decreased during the evolution. All runs converged to interesting robot structures capable of progressing in water (Table 1). The minimal number of generations required for the convergence is 128 (B5) and the average number of generation is 860.

Description of the most efficient robots. The only two robots (A5 and B3) that are identical among all runs are, interestingly, also the most efficient, the simplest and the smallest of all. As they are the only two robots that have the same shape and as they are the most efficient, they probably correspond to a good optimum in the search space. They consist of 5 modules forming two limbs of unequal sizes (fig. 4 top left). All the oscillators are activated although this can appear useless since only the joint number 2 is really generating thrust. However, all contribute to the general behavior, and the fact that the limbs oscillate on themselves reduces their rubbing, and helps the locomotion.

Diversity. Except for the best solutions of runs A5 and B3 mentioned above, all runs evolved to different types of robots. The number of parameters to be optimized and their relatively large intervals of values, make the search space very large. With the ten runs presented here, we thus explore only a small part of the search space. That, and the topology of the search space which might have many local optima, probably explains the diversity of the results.

Table 1. Table of the ten runs. *Velocity* means : Average velocity on the XY plane of the best robot in [m/min] calculated during 15 seconds after 15 seconds for stabilization. *INM* means Initial Number of Modules and specifies the size in terms of modules of the robots of the first generation. *FNM* means Final Number of Modules and is the final size in terms of modules of the evolved robots.

Set	ID	INM	FNM	Fitness Max.	Fitness Av.	Velocity
A	1	5	6	4.81	4.76	9.08
	2	5	6	4.08	3.98	6.8
	3	5	5	5.32	5.22	10.03
	4	5	6	4.8	4.71	9.42
	5	5	5	6.3	6.23	10.58
B	1	10	8	4.6	4.53	9.13
	2	10	9	4.63	4.54	9.14
	3	10	5	5.46	5.41	10.50
	4	10	9	3	2.99	6.12
	5	10	8	2.25	2.2	4.79

Evolution of the number of modules. The five runs of the set A did not add or remove large number of modules to or from robots compared to the initial populations and to a lesser extent it is also the case for runs of the set B. Simple and effective solutions are quickly found by the GA and the addition or the withdrawal of modules is almost never done. It is primarily the neural network which is optimized. This "inertia" of the size is probably explained by the fact that adding or removing a module constitutes a significant perturbation of both the dynamics of the body and of the global neural network.

Symmetry. In nature, the majority of animals have an axis of symmetry. For an efficient, controllable and rectilinear locomotion it seems to be necessary. However, we made the choice not to force this symmetry and to see if it appeared spontaneously. That was not the case here maybe because for having and keeping symmetrical structures two mutations must appear at the same time and symmetrically which is unlikely (see [21] for a related study). To go straight in spite of this handicap, all robots follow spiral trajectories turning on themselves. As the center of this spiral is a line, the robots go as straight as possible with respect to the fitness function.

Evolution of the BS. The values of the left and right BS are evolved with the robots and can be varied between 0.2 and 1.13 (cf the neural oscillator). Half of the robots converge to the maximum value for the two inputs. Indeed, to swim as fast as possible it seems to be important to have high frequencies and amplitudes of the movements. The others robots have different values for the left and right BS.

In vertebrate animals, it is known that increasing the tonic drive from the brainstem increases the speed of locomotion. We tested if the robots which we evolved reacted in same the manner. We noted that speed indeed increased or decreased according to BS (cf Fig 4 top right) as awaited, although that must

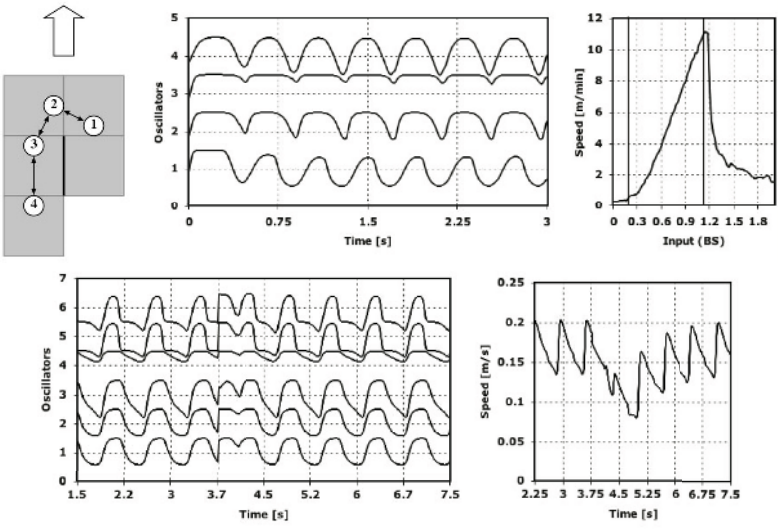


Fig. 4. **Top left**: Robot evolved by the run A5 (same shape as B3). **Top middle**: Neural activity of the robot A5. Each curve corresponds to the difference between left and right motoneuron of one oscillator. The four oscillators are synchronized and appear to be able to generate a lot of different signals. **Top right**: Influence of the tonic input (BS) on the velocity of the robot A5. The vertical lines determine the region in which the velocity increases monotonically with the BS. **Bottom left**: Neural activity of the robot B2. Each curve corresponds to the difference between left and right motoneuron of one oscillator. At 3.75 seconds, a random perturbation is applied to membrane potentials. **Bottom right**: Evolution of the speed of the center of mass of the robot. The oscillations are due to the periodic flutters of the limbs.

be moderated because the trajectory also changes, i.e. if the robot goes straight with maximum and symmetrical input values, it happens that the robot does not swim straight anymore when we decrease symmetrically.

Resistance to perturbations. We tested the robustness of our robots to perturbations by perturbing the membrane potentials of all neurons and setting them to random values. The neural activity and the speed of the robot B2 is shown on figure 4 on the bottom. The neural activity rapidly recovers from the perturbation and returns to steady state after 2 oscillations. The velocity also recovers although it takes a little bit more because the robot has to struggle against the water inertia. This robustness against perturbations is one of the interesting features of using CPG models for locomotion.

3 Conclusion

This article presented first results of a project in underwater modular robotics, called Neubots. Body structures and neural controllers based on central pattern generators were co-evolved for efficient underwater locomotion. The main results

are the design of neural oscillators linked together by synaptic spreading capable of producing robust signals for locomotion. The controllers can adjust the speed of locomotion by the modulation of simple signals and quickly recover from random perturbations in the neural activity.

Acknowledgements. We are grateful to A. Guignard, T. Bertolote and V. Hentsch for their implication in the design of the modules. Thanks also to P. Duerr for his collaboration on the design and implementation of the simulator. We acknowledge support from the Swiss National Science Foundation for a Young Professorship grant to Auke Ijspeert.

References

1. D. Floreano, A. J. Ijspeert, T. Bertolote, and C. Mattiussi. Neubots: The robotics legacy of John von Neumann. Technical report, submitted for publication, Laboratory of Intelligent Systems, EPFL, 2005.
2. T. Bertolote and V. Hentsch, Design and prototyping of an underwater modular robot, Unpublished Master Thesis, Laboratory of Intelligent Systems, EPFL, 2004.
3. J. vonNeumann. *Theory of Self-reproducing Automata*. University of Illinois Press, Urbana, IL, 1966. Edited and completed by A. W. Burks.
4. W. McCulloch and W. Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.
5. Williams, T. Phase coupling by synaptic spread in chains of coupled neuronal oscillators. *Science*, 258, 662-665, 1992
6. S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji. M-tran: Self-reconfigurable modular robotic system. *IEEE/ASME Transactions on Mechatronics*, 7:431–441, 2002.
7. G. Konidaris, T. Taylor, and J. Hallam. Hydrogen: Automatically generating self-assembly code for hydron units. In *Proceedings of the Seventh International Symposium on Distributed Autonomous Robotic Systems (DARS04)*, Berlin, 2004.
8. Sims, K.: Evolving 3D Morphology and Behavior by competition. Artificial Life IV Proceedings, ed. by R. Brooks & P. Maes, MIT Press, pp 28-39, 1994.
9. Ekeberg, Ö.: A combined neuronal and mechanical model of fish swimming. *Biological Cybernetics*, 69, 363-374, 1993.
10. Grillner, S.: Neurobiological bases of rhytmic motor acts in vertebrates. *Science*, New Series, Vol. 228, No. 4696, pp 143-149, Apr. 12 1985.
11. Kamimura, A., Kurokawa, H., Yoshida, E., Tomita, K., Murata, S., Kokaji, S.: Automatic Locomotion Pattern Generation for Modular Robots. *Proceedings of the 2003 IEEE International Conference on Robotics & Automation, 2003*
12. Yoshida, E., Murata, S., Kamimura, A., Tomitd, K., Kurokawa, H., Kokaji, S.: Evolutionary Synthesis of Dynamic Motion and Reconfiguration Process for a Modular Robot M-TRAN. *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2003.
13. Fukuoka, Y., Kimura, H., Cohen, A. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. *The International Journal of Robotics Research*, 3-4, pp 187-202, 2003
14. Ijspeert, A. J.: A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander. *Biol. Cybern.*, Vol. 84:5, pp 331-348, 2001.

15. M. Komosinski, S. Ulatowski, Framestics: Towards a simulation of a nature-like world, creatures and evolution. *ECAL 1999*, pp. 261-265, 1999
16. H. Lipson, J.B. Pollack, Automatic design and manufacture of robotic lifeforms, in *Nature*, 406:974-978, 2000.
17. J.C. Bongard, R. Pfeifer, Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontongey, in *Genetic and Evolutionary Computation Conference*, p. 829-836, 2001.
18. Marbach, D., Ijspeert, A. J.: Co-evolution of Configuration and Control for Homogenous Modular Robots. *Proceedings of the Eighth Conference on Intelligent Autonomous Systems (IAS8)*, pages 712-719. IOS Press, 2004.
19. Smith, R.: Open Dynamics Engine (ODE), webpage: http://www.ode.org/.
20. GAlib, webpage: http://lancet.mit.edu/ga/
21. Bongard, J.C. and C. Paul, Investigating Morphological Symmetry and Locomotive Efficiency using Virtual Embodied Evolution, in *From Animals to Animats: The Sixth Int. Conference on the Simulation of Adaptive Behaviour*, pp. 420-429, 2000.

CoEvolutionary Incremental Modelling of Robotic Cognitive Mechanisms

Michail Maniadakis and Panos Trahanias

Inst. of Comp. Science, Foundation for Research and Technology-Hellas,
71110 Heraklion, Crete, Greece
Department of Computer Science, University of Crete, 71409 Heraklion, Crete, Greece
{mmaniada, trahania}@ics.forth.gr

Abstract. Recently, brain models attempt to support cognitive abilities of artificial organisms. Incremental approaches are often employed to support modelling process. The present work introduces a novel computational framework for incremental brain modelling, which aims at enforcing partial components re-usability. A coevolutionary agent-based approach is followed which utilizes properly formulated neural agents to represent brain areas. A collaborative coevolutionary method, with the inherent ability to design cooperative substructures, supports the implementation of partial brain models, and additionally supplies a consistent method to achieve their integration. The implemented models are embedded in a robotic platform to support its behavioral capabilities.

1 Introduction

The long-term vision of developing artificial organisms with mammal-like cognitive abilities, has recently given impetus in brain modelling studies. The brain of mammals consists of several interconnected modules with different functionalities [5], which implies that models with distributed architecture should be designed. Recently, we proposed a novel coevolutionary method [6] [8], to design distributed partial brain models. Specifically, neural network agents are coevolved by distinct species (populations) emphasizing both their autonomy and cooperability with the remaining structures.

Additionally, incremental brain modelling approaches have been proposed [9,15,13]. However, the computational structures employed by the proposed incremental approaches suffer in terms of scalability, and can not be used widely as a brain modelling computational framework. This is because substructures are originally designed to handle a specific amount of incoming information. Thus, by performing incremental modelling steps, the structures are difficult to operate successfully because new modules are integrated, and additional information volume is projected on them. Furthermore, no optimization method is employed to support the incremental modelling process.

The coevolutionary method matches adequately the incremental modelling processes due to its inherent ability to integrate distributed structures. In the present work, we propose a brain modelling method focusing on the integration

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 200–209, 2005.

of partial models in gradually more complex ones. Specifically, in order to eliminate the problem of existing computational models employed by incremental processes, we utilize neural modules with internal dynamics, which self-adapt their performance as new structures are integrated on top of them. Intermediate link modules are employed which are connected on the key points of existing structures, to properly modulate their performance. Furthermore, a coevolutionary optimization method facilitates the incremental process, offering a consistent mechanism to support the reusability of substructures. The proposed approach is assessed by embedding the implemented brain model in a robotic platform, to furnish it with cognitive capabilities.

The rest of the paper is organized as follows. In the next section we present the proposed computational framework consisting of the agent structures employed to represent partial brain areas, and the collaborative coevolutionary scheme which specifies the computational details of brain models. Computational experiments which follow the proposed framework to design a partial brain model are presented in section 3. Specifically, we demonstrate the implementation of a computational model simulating posterior parietal cortex - prefrontal cortex - primary motor cortex - spinal cord interactions in a delayed response task. Finally, conclusions and suggestions for further work are drawn in the last section.

2 Computational Framework

The proposed computational framework is inspired by the biological prototype, while at the same time serves the special needs of incremental modelling. Specifically, brain areas are modelled by flexible neural network agents. Similarly to a phylogenetic process, an evolutionary approach is employed to specify the computational details for each agent [14]. Instead of using a unimodal evolutionary process, a collaborative coevolutionary method is utilized to support neural agent specification, offering enhanced search abilities of partial brain elements [11]. In the following, we present in turn the computational structures, and the coevolutionary approach.

2.1 Computational Model

We implement two different neural agents, to supply a general computational framework: (i) a cortical agent to represent brain areas, and (ii) a link agent to support information flow across cortical modules. Their structures are an extension of those presented in [8], [7].

Link Agent. The structure of link agent is appropriately designed to support connectivity among cortical modules. Using the link agent any two cortical modules can be connected, simulating the connectivity of brain areas.

Each link agent is specified by the projecting axons between two cortical agents (Fig 1(a)). Its formation is based on the representation of cortical agents by planes with excitatory and inhibitory neurons (see below). Only excitatory neurons are used as outputs of the efferent cortical agent. The axons of projecting

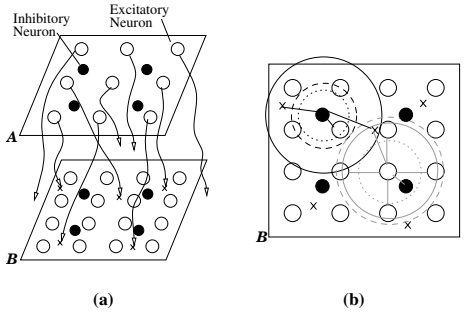


Fig. 1. Schematic representation of the computational model. Part (a) illustrates a link agent which supports information flow from cortical agent A to B. Only the active projections are represented with an × on their termination. Part (b) illustrates synapse definition in cortical agent B. Neighborhood radius for i) afferent axons is illustrated by a solid line, for ii) neighboring excitatory neurons by a dashed line, and for iii) neighboring inhibitory neurons by a dotted line. Sample neighborhoods for excitatory neurons are illustrated with grey, while neighborhoods for inhibitory neurons are illustrated with black.

neurons are defined by their (x, y) coordinates on the receiving plane. Cortical planes have a predefined dimension and thus projecting axons are deactivated if they exceed the borders of the plane. The proposed link structure facilitates the connectivity of sending and receiving cortical agents supporting their combined performance.

Cortical Agent. Each cortical agent is represented by a rectangular plane. A cortical agent consists of a predefined population of excitatory and inhibitory neurons, which all follow the Wilson-Cowan model with sigmoid activation as it is described in [8]. Both sets of neurons, are uniformly distributed, defining an excitatory and an inhibitory grid on the cortical plane. On the same plane there are also located the axon terminals from the efferent projected cortical agents.

All neurons receive input information either from i) projecting axons, or ii) excitatory neighboring neurons, or iii) inhibitory neighboring neurons. The connectivity of neurons follows the general rule of locality. Synapse formation is based on circular neighborhood measures. A separate radius for each of the three synapse types, defines the connectivity of neurons. This is illustrated graphically in Fig 1(b), which further explains Fig 1(a). All excitatory neurons share common neighborhood measures. The same is also true for all inhibitory neurons.

The performance of cortical agents is adjusted by the experiences of the artificial organism, obtained through environmental interaction, similar to epigenetic[1] learning [2]. To enforce experience based subjective learning, each set of synapses is assigned a Hebbian-like learning rule defining the self-organization internal dynamics of the agent. We have implemented a pool of 10 Hebbian-like rules that can be appropriately combined to produce a wide range of function-

[1] Epigenesis here, includes all learning processes during lifetime.

alities. The employed learning rules are the union of those employed in [3], [6], and thus they are omitted here due to space limitation. Agents plasticity allows synaptic adjustments at run-time based on environmental experience. The most common, but harder to evolve, alternative of genetically-encoded synaptic strengths, results to a rather unmanageable problem complexity.

2.2 Collaborative Coevolution

An evolutionary process determines the self-organization dynamics of partial brain structures, enforcing the emergence of valuable behaviors during lifetime. However using a unimodal evolutionary approach, it is not possible to explore effectively partial solutions, which correspond to brain substructures. Coevolutionary algorithms have been recently proposed to facilitate exploration in problems consisting of many decomposable subcomponents (e.g [10,11]). Distinct species (populations) are employed to estimate solutions for each partial component of the problem. Accordingly, increased search competencies are inherently available in coevolutionary algorithms, while the special characteristics of substructures can be also taken into account. Recently, we introduced the usage of collaborative coevolution for the design of partial brain models [6] [8], while in the present study we demonstrate that this approach can also serve the incremental modelling process.

Specifically, a two level collaborative coevolutionary scheme is employed. The species representing distinct elements of the composite system are evolved independently at the lower level. Additionally, an evolutionary process performs at a higher level, to select the appropriate individuals from each species that cooperatively are able to construct a good composite solution. Thus the parameter space is segmentally searched in the lower level by evolved species, while at the same time, the high level evolutionary process searches within species to identify the best collaborator schemes.

We employ two kinds of species encoding the configurations of either a Primitive agent Structure (PS) or a Coevolved agent Group (CG). PS species specify partial elements, encoding the exact structure of either cortical or link agents. A CG consists of a group of cooperating PSs with common objectives. Thus, CGs specify configurations of partial solutions by encoding individual assemblies of cortical and link agents (see Fig 2).

A general purpose genotype is employed for both the lower level evolution of species, and the higher-level collaborator selection process. According to that, each individual is assigned an identification number and encodes two different kinds of variables. The first kind is allowed to get a value from a set of unordered numbers, e.g. {1,5,7,2}, with the ordering of the elements being of no use. These variables are called SetVariables. The second kind of variables is allowed to get a value within a range of values, e.g. [0,1]; therefore, they are called RangeVariables. The computational details of PS (either cortical or link) and CG structures can be easily mapped to the genotype, following a process very similar to the one described in [8]. This is omitted here due to space limitations.

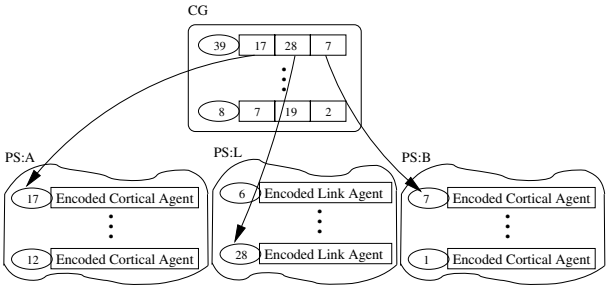


Fig. 2. A schematic overview of the coevolutionary process. CG is represented by a rounded box, while PSs are represented by a free shape. Identification numbers are represented with ovals.

In order to test the performance of individuals, the population at the higher level is accessed. The parameter values at CG-level guide collaborator selection among PS species (Fig 2). Then collaborators are combined to form the proposed solution which is further tested. During fitness assignment, CG-individuals are assigned a fitness value f, representing how good is the solution formed by the selected collaborators. Individuals of the coevolved PS-species at the lower level are assigned the maximum of the fitness value achieved by all the solutions formed with their membership. Thus an individual of the lower level species is assigned the value $f = max\{f_i\}$ where f_i is the fitness value of the i-th solution formed with the collaboration of the individual under discussion.

Evolutionary steps are performed based on the standard evolutionary operators. First, individuals of each species are sorted according to their fitness values. Then, a predefined percentage of individuals are crossed over. An individual selects its mate from the whole population, based on their accumulative probabilities. Finally, mutation is performed in a small percentage of the resulted population. Genetic operators are applied in both levels in the same way.

2.3 Discussion

The plasticity of agent structures, which stems from the assignment of learning rules, constitute a key feature of the proposed computational model. Specifically, it facilitates the incremental modelling process by adjusting the performance of each module to various circumstances of incoming information, enforcing the reusability of substructures. This is a novel feature of our approach since, although neural structures with self-organization characteristics are widely used in many different domains, their suitability on modelling incrementally distributed systems has not been studied before.

It should be noted that coevolution is not the only methodology to approach incremental modelling. Other optimization processes (e.g. unimodal evolution) would theoretically be able to support the incremental process. However, coevolution offers many advantages in terms of searching effectively partial solutions,

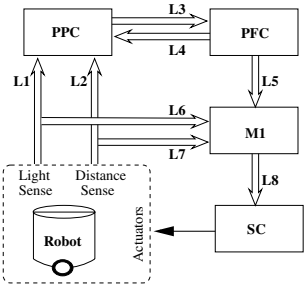


Fig. 3. A schematic overview of the Primary Motor Cortex model. Cortical agents are illustrated with blocks, while link agents are illustrated with a double arrow.

because it is originally designed to work with substructures instead of the composite solution. As a result coevolution matches adequately to the agent-based distributed brain modelling. This has been illustrated in [6], [8] where one-step coevolutionary processes are employed to design brain models consisting of independent but cooperable substructures with distinct functional goals.

Furthermore, the proposed coevolutionary scheme can be hierarchically organized supporting the concurrent optimization of many substructures in one-step [7]. The hierarchical approach can be used also to overcome the well known problem of incremental modelling where the constraints imposed by the structure of initial models can be too hard, harming the forthcoming incremental steps. By commencing a hierarchical coevolutionary process which loads the results of the first incremental steps it is possible to further optimize them considering also the needs of the new components. As a result "single-step" and "incremental" processes can support each other, performing in a complementary way.

3 Results

The effectiveness of the proposed approach is illustrated on the design of a partial brain computational model, which simulates posterior parietal cortex (PPC) - prefrontal cortex (PFC) - primary motor cortex (M1) - spinal cord (SC) interactions, emphasizing on working memory (WM) usage (Fig 3). We note that the proposed model does not aim to be a detailed replica of the biological prototype (e.g. premotor areas are not represented), but it serves as a guide on how incremental coevolution can be employed to support brain modelling.

Several biological experiments highlight the behavioral organization of these brain areas. They are based on delayed response (DR) tasks which require to retain memory relative to a sample cue for a brief period, in order to decide upon future behavioral response (e.g. [12]). M1 encodes primitive motor commands which are expressed to actions by means of SC. PPC-PFC reciprocal interaction operates in a higher level encoding WM, to develop plans regarding future actions. PFC activation is then passed to M1 which modulates its performance

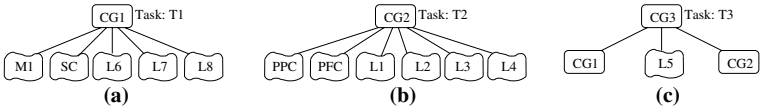


Fig. 4. A schematic overview of the incremental coevolutionary process employed to design the model of Fig 3. Part (a) illustrates the process employed to design the model of M1-SC interaction, part (b) illustrates the process designing the model of PPC-PFC interaction, and part (c) illustrates the coevolutionary process which serves their integration.

accordingly. Thus, the higher level orders specify the expressed actions, aiming at the accomplishment of the DR-task.

The interactions of the brain areas under discussion are modelled incrementally. The process starts by two coevolutionary processes implementing separate computational models of both M1-SC and PFC-PPC interactions. These two models are further integrated by means of another coevolutionary process operating on top of them. Both partial and composite models are embedded on a simulated mobile robot to furnish it with cognitive abilities and prove the validity of results. We employ a two wheeled robotic platform equipped with 8 uniformly distributed distance and light sensors.

Three tasks (adjusted to the needs of robotic applications) are properly specified, in order to demonstrate the effectiveness of the computational procedure. The first task $T1$, accounts for primitive motion abilities without purposeful planning. For mobile robots, a task with the above characteristics is wall avoidance navigation. Since M1-SC are the brain modules which serve basic motor commands, and they are operative after lesion of the higher level structures [5], it is assumed that they are relevant for the accomplishment of wall avoidance navigation.

M1-SC interactions are modelled by means of a coevolutionary process illustrated in (Fig 4(a)). The success of wall avoidance task is evaluated by the fitness function:

$$F_1 = \left(\sum_M (sl + sr - 1) * (1.0 - p^2)\right) * \left(1 - \frac{2}{M}\left|\sum_M \frac{sl - sr}{sl * sr}\right|\right)^3 * \left(1 - 2\sqrt{\frac{B}{M}}\right)^3$$

where we assume that the robot is tested for M steps, sl, sr are the instant speeds of the left and right wheel, p is the maximum instant activation of distance sensors, and B is the total number of robot bumps. The first term seeks for forward movement far from the walls, the second supports straight movement without unreasonable spinning, and the last term minimizes the number of robot bumps on the walls. A sample result is illustrated in Fig 5(a).

The development of WM-like performance specifies the second task $T2$. Working memory (WM) is the ability to hold and manipulate goal-related information to guide forthcoming actions. The PFC and PPC are the brain structures most closely linked to WM [1]. Thus PPC-PFC are responsible for WM develop-

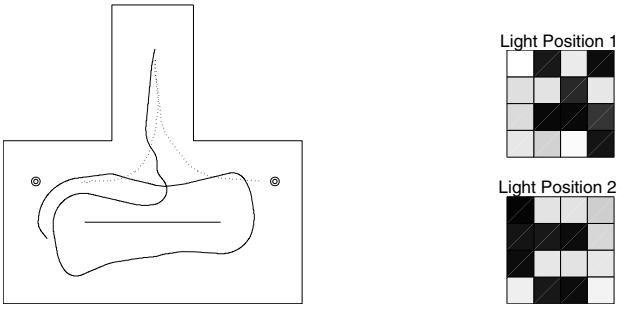


Fig. 5. Part(a) illustrates robot performance on wall avoidance navigation (solid line), and the delayed matching-to-sample task (dotted line). Targets are illustrated with double circles. Part(b) illustrates the average activation of excitatory neurons at PFC. Dark activation values indicate that the cell remain active during all the observed period, while light values indicate low activity in the same period. Evidently, each side of light cue presence is encoded by a different activation pattern.

ment in the proposed computational model. In the present experiment, a light cue is presented in the left or right side of the robot. WM performance aims at persistent PFC activity, related each time to the respective side of light cue.

Two different states l, r are defined, associated to the left or right side of light source appearance. For each state, separate activation-averages over the time of M simulation steps, a_j, are computed, with j identifying excitatory neurons of PFC agent. The formation of WM related to the side of light cues is evaluated by measuring the persistency of activation in PFC:

$$F_2 = \frac{1}{2}\left(\frac{v_l}{m_l} + \frac{v_r}{m_r}\right) * \min\left\{ \sum_{j, a_j^l > a_j^r} \left(a_j^l - a_j^r\right), \sum_{j, a_j^r > a_j^l} \left(a_j^r - a_j^l\right) \right\}$$

where m_l, v_l, m_r, v_r are the mean and variance of average activation at the respective states. The first term seeks for consistent PFC activation, and the second supports the development of a distinct set of active neurons for each state. A sample result is illustrated in Fig 5(b).

When the first two processes are completed, a third coevolutionary scheme commences to design the intermediate link structure which integrates the performance of the two partial models in a compound one. Following the hierarchy of motor brain areas in mammals, the memory held by PFC activation modulates M1 performance to develop goal directed motion [5,4]. The successful interaction of substructures is demonstrated by means of a delayed response (DR) task. Specifically, a light cue is presented on the left or right side of the robot. The robot has to move at the end of a corridor memorizing the side of sample cue appearance, and then make a choice related to 90^o turn left or right, depending on the side of light cue presence.

A target location is defined in each side of the corridor depending on the position of the initial light cue (Fig 5). The robot has to approximate the target location without bumping on the walls. The successful approximation to the target location is estimated by:

$$G = \left(1 + 3.0 * \left(1 - \frac{d}{D}\right)\right)^3 * \left(1 - 2\sqrt{\frac{B}{M}}\right)^2$$

where d is the minimum euclidian distance between the target and the robot, D is the euclidian distance between the target and the starting location of the robot, and B is the total number of robot bumps. The accomplishment of $T3$ is evaluated by means of two subtasks testing separately the right or left turn of the robot for the respective positions of the light cue, employing each time the appropriate target location:

$$F_3 = G^l * G^r$$

The third hierarchical scheme performs on the results of the previous two processes evolving additionally the link agent $L5$ to support their connectivity (Fig 4(c)). The ten best individuals of $CG1$ and $CG2$ species are used as indicative partial structures to form a basis for the construction of the global model. Thus, only two species are evolved. The lower level species encoding the structure of $L5$ link agent, and $CG3$ species which choose the appropriate collaborator assembly. A sample result is illustrated in Fig 5(a).

It is easily observed that the self-organization dynamics of M1-SC structures allow the modulation of their performance according to the higher level orders. Thus, their functionality is adapted successfully from wall avoidance to goal reaching. As a result, the proposed computational framework achieves the construction of a new complex model from simple components, while the behavioral repertory of the robot is enriched.

4 Conclusions

In the present work we proposed an incremental computational framework to support brain modelling efforts. It follows an agent based approach able to simulate the distributed organization of brain areas. The employed cortical agents exhibit self-organization dynamics which serve both the experience-based learning, and the incremental modelling process by adjusting the performance of agents on circumstances with different amounts of incoming information. The employed link agents are properly formulated to connect the key sending and receiving points of cortical structures in order to achieve their integrated performance. Furthermore, the coevolutionary design approach, which matches the distributed architecture of the computational model, facilitates the integration of substructures in composite ones.

The proposed computational framework exploits the inherent ability of coevolution to integrate partial structures, exhibiting the following advantages:

- it offers a systematic methodology to facilitate incremental brain modelling process by gradually adding new coevolved species to represent brain areas,
- it supports both individual and cooperative characteristics of brain areas,
- it supports the construction of complex behaviors from simple components.

Consequently, the proposed method can potentially support large-scale brain modelling tasks and the development of competent artificial cognition mechanisms. Further work is currently underway, to investigate the suitability of our approach in large scale brain modelling tasks and the endowment of cognitive abilities to artificial organisms.

References

1. A. Compte, N. Brunel, P.S. Goldman-Rakic, X.-J. Wang. Synaptic mechanisms and network dynamics underlying spatial working memory in a cortical network model. Cerebral Cortex, 10(1):910-923, 2000.
2. R.M.J. Cotterill. Cooperation of the basal ganglia, cerebellum, sensory cerebrum and hippocampus: possible implications for cognition, consciousness, intelligence and creativity. Progress in Neurobiology, 64(1):1-33, 2001.
3. D. Floreano, J. Urzelai. Evolutionary robots with on-line self-organization and behavioral fitness. Neural Networks, 13:431-443, 2000.
4. J.M. Fuster. Executive frontal functions. Experimental Brain Research, 133:66-70, 2000.
5. E. R. Kandel, J.H. Schwartz, T. M. Jessell. Principles of Neural Science. Mc Graw Hill, 2000.
6. M. Maniadakis, P. Trahanias. Evolution tunes coevolution: modelling robot cognition mechanisms. In proc. of Genetic and Evolut. Comput. Conference (GECCO-2004), pp. 640-641, 2004.
7. M. Maniadakis, P. Trahanias. A Hierarchical Coevolutionary Method to Support Brain-Lesion Modelling. In proc. of Int. Joint Conference on Neural Networks, (IJCNN-2005), 2005.
8. M. Maniadakis, P. Trahanias. Modelling Brain Emergent Behaviors Through Coevolution of Neural Agents. accepted for publication, Neural Networks journal.
9. G. Metta, F. Panerai, R. Manzotti, G. Sandini. Babybot: an artificial developing robotic agent. In proc. of SAB 2000.
10. D.E. Moriarty, R. Miikkulainen. Forming Neural Networks Through Efficient and Adaptive Coevolution. Evolutionary Computation, 5(4):373-399, 1997.
11. M. Poter, K. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evolutionary Computation, 8:1-29, 2000.
12. M.E. Ragozzino, R.P. Kesner. The role of rat dorsomedial prefrontal cortex in working memory for egocentric responces. Neuroscience Letters, 308:145-148, 2001.
13. R.G. Reilly, I. Marian. Cortical Software Re-Use: A Computational Principle for Cognitive Development in Robots. In proc. ICDL 2002.
14. E.T. Rolls, S.M. Stringer. On the design of neural networks in the brain by genetic evolution. Progress in Neurobiology, 61:557-579, 2000.
15. B. Scassellati. Theory of Mind for a Humanoid Robot. Autonomous Robots, 12(1):13-24, 2002.

A Dynamical Systems Approach to Learning: A Frequency-Adaptive Hopper Robot

Jonas Buchli, Ludovic Righetti, and Auke Jan Ijspeert

Biologically Inspired Robotics Group,
Ecole Polytechnique, Fédérale de Lausanne, (EPFL)
jonas.buchli@epfl.ch
http://birg.epfl.ch

Abstract. We present an example of the dynamical systems approach to learning and adaptation. Our goal is to explore how both control and learning can be embedded into a single dynamical system, rather than having a separation between controller and learning algorithm. First, we present our adaptive frequency Hopf oscillator, and illustrate how it can learn the frequencies of complex rhythmic input signals. Then, we present a controller based on these adaptive oscillators applied to the control of a simulated 4-degrees-of-freedom spring-mass hopper. By the appropriate design of the couplings between the adaptive oscillators and the mechanical system, the controller adapts to the mechanical properties of the hopper, in particular its resonant frequency. As a result, hopping is initiated and locomotion similar to the bound emerges. Interestingly, efficient locomotion is achieved without explicit inter-limb coupling, i.e. the only effective inter-limb coupling is established via the mechanical system and the environment. Furthermore, the self-organization process leads to forward locomotion which is optimal with respect to the velocity/power ratio.

1 Introduction

Nonlinear dynamical systems are a promising approach both for studying adaptive mechanisms in Nature and for devising controllers for robots with multiple degrees of freedom. Indeed, nonlinear dynamical systems can present interesting properties such as attractor behavior which can be very useful for control, e.g. the generation of rhythmic signals for the control of locomotion. However, controllers for engineering applications usually need to be tailor-made and tuned for each application. This is in contrast to Nature where multiple adaptive mechanisms take place to adjust the controller (the central nervous system) to the body shape, and vice-versa. For the control of locomotion for instance, there are mechanisms to adapt the locomotor networks to changing body properties (e.g. due to growth, aging, and/or lesions) during the life time of an individual, and this greatly increases its survival probability.

In order to endow robots with similar capabilities, we are investigating the possibility to construct adaptive controllers with nonlinear dynamical systems. This is achieved by letting the parameters of the system change in function

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 210–220, 2005.

of the systems behavior, and, therefore, also in function of external influences. Thus, we aim at designing systems in which learning is an integral part of the dynamical system, not a separate process, in contrast to many approaches in artificial neural networks and other fields.

In an earlier study [4] we presented an *adaptive frequency oscillator* as a controller for a simple locomotion system. The motivation to use an adaptive frequency oscillator was to deploy a controller that is able to adapt to "body"-properties, i.e. properties of the mechanical system. In this case the body property to be adapted to is the resonant frequency of the mechanical system. The presented approach is especially useful when the mechanical properties of the body are not known or changing. In such cases, properties such as the resonant frequencies and similar are not directly accessible, but have to be inferred by some sort of measurement.

In this paper we pursue further the idea of the adaptive frequency oscillator used as an adaptive locomotion controller (cf. [4]). First, we will present how the adaptive frequency oscillator can learn the frequencies of arbitrary rhythmic input signals. The main interesting features of the adaptive oscillator are (1) that it can learn the frequencies of complex and noisy signals, (2) that it does not require any pre-processing of the signal, and (3) that the learning mechanism is an integral part of the dynamical system.

Then, we will present a more complex and realistic example of a robot that is capable of hopping, namely a 4-DOF spring-mass hopper with an adaptive controller based on the adaptive frequency oscillators. Spring-mass systems have been widely used to study fundamental aspects of locomotion [3, 11] and several robots based on this concept have been presented [17, 8, 5]. In [12] the mechanical stability of spring-mass systems is discussed. Recently, robots with legs including elastic elements have been presented [10, 13, 14]. Coupled oscillators have been extensively studied for locomotion control [7, 20, 19, 10]. However, usually the structure and parameterization of these controllers are fixed by heuristics or are adapted with algorithms which are not formulated in the language of dynamical systems. One exception is [16] where learning is included in the dynamical system. The results are, however, for many coupled phase oscillators and no direct application example is given. Another exception is [6] where an adaptation of the stride period is investigated, with a discrete dynamical system.

In our contribution, thanks to the adaptive mechanisms, the controller tunes itself to the mechanical properties of the body, and generates efficient locomotion. As we will see, the system, albeit its simplicity, shows a rich and complicated emergent behavior. In particular, efficient gait patterns are evolved in a self-organized fashion, and are quickly adjusted when body properties are changed. Interestingly, the emergent gaits are optimal with respect to the velocity/power ratio.

2 Adaptive Frequency Oscillators

In this section we introduce our adaptive frequency Hopf oscillator, and will show its behavior under non-harmonic driving conditions.

The adaptive frequency Hopf oscillator is described by the following set of differential equations. We introduce it in the Cartesian coordinate system (Eqs. 1–3) as this allows an intuitive understanding of the additive coupling. In order to understand convergence and locking behavior it is also convenient to look at the oscillator in its phase, radius coordinate system which in the case of the Hopf oscillator, having a harmonic limit cycle, coincides with the representation in polar coordinates (Eqs. 4–6).

$$\dot{x}_h = (\mu_h - r^2)x_h - \omega_h y_h + cF_x(t) \quad (1)$$

$$\dot{y}_h = (\mu_h - r^2)y_h + \omega_h x_h \quad (2)$$

$$\dot{\omega}_h = -\frac{1}{\tau_h}\frac{y}{r}cF_x(t) \quad (3)$$

$$\dot{r}_h = (\mu_h - r^2)r + \cos(\phi_h)cF_x(t) \quad (4)$$

$$\dot{\phi}_h = \omega_h - \frac{1}{r}\sin(\phi_h)cF_x(t) \quad (5)$$

$$\dot{\omega}_h = -\frac{1}{\tau_h}\sin\phi_h cF_x(t) \quad (6)$$

where x_h, y_h are the states of the oscillator, ω_h is its intrinsic frequency, $r = \sqrt{x_h^2 + y_h^2}$ and $F_x(t)$ is a perturbing force (the subscript h distinguishes the variables of the Hopf oscillator from variables in the mechanical system). If $F_x(t) = 0$, this system shows a structurally stable, harmonic limit cycle with radius $r = \sqrt{\mu}$ for $\mu > 0$. It can be shown [18] that such an oscillator adapts to frequencies present in a rhythmic input signal. In the case of a harmonic signal $F_x(t) = \sin(\omega_F t)$ this means ω_h is evolving toward ω_F. If the input signal has many frequency components (e.g. square signal) the final value of ω_h is dependent on the initial condition $\omega_h(0)$. The size and boundaries of the basins of attraction are proportional to the energy content of the frequency component constituting the basin of attraction, see [18] for further discussion of the convergence properties. Our adaptive frequency oscillators have many nice features which makes them useful for applications and a good example for the dynamical systems approach to learning: 1) no separation of learning substrate and learning algorithm, 2) learning is embedded into the dynamics, 3) no preprocessing needed (e.g. no extraction of phase, FFT, nor setting of time windows), 4) work well with noisy signals, 5) robust against perturbation, 6) they possess a resonant frequency and amplification properties.

Now we shall present a few representative results from numerical integration, to show the correct convergence of the adaptive frequency oscillator. First, we show the convergence for a harmonic perturbation $F_x(t) = \sin(\omega_F t)$. As we are interested to show that $\omega_h \to \omega_F$, we use $\omega_d = \omega_h - \omega_F$ and $\phi_d = \phi_h - \phi_F$ to plot the results. For all simulations $\tau = 1$, $c = 0.1$, $\omega_d(0) = 1$, $\phi_d(0) = 0$ and $r_h(0) = 1$. We present results of the integration of the system Eqs. 4-6. In Fig. 1 the behavior of variables ϕ_d and ω_d is depicted. In Fig. 1(b), we present the phase plot of the system for the harmonic perturbation. Clearly visible is that the system is evolving towards a limit set. The limit set corresponds to the phase locked case $\phi_d \leq const$ and the frequency has adapted so that $\omega_d \approx 0$. Since we want to be sure that the convergence works for a wide range of input signals (as in the case when the oscillator is coupled with the mechanical system) we show then results with general nonharmonic perturbation by general T_F-periodic functions $f(t, \omega_F)$, $T_F = \frac{2\pi}{\omega_F}$ (Fig. 2). The system was subjected to the following driving signals: (a) Square Pulse Signal

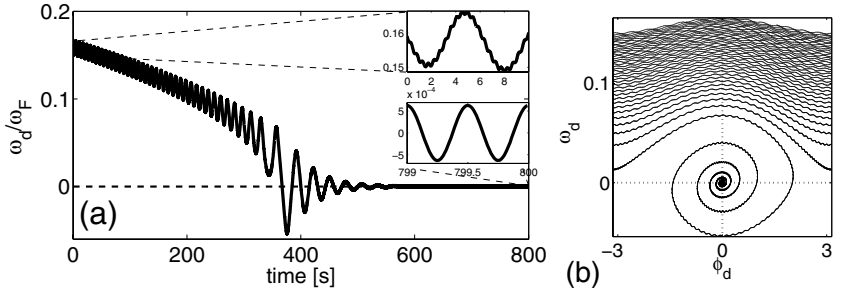


Fig. 1. (a) Integration of the System Eqs. 1-3 (b) corresponding phase plot, in which the frequency adaptation and the phase locking can be seen.

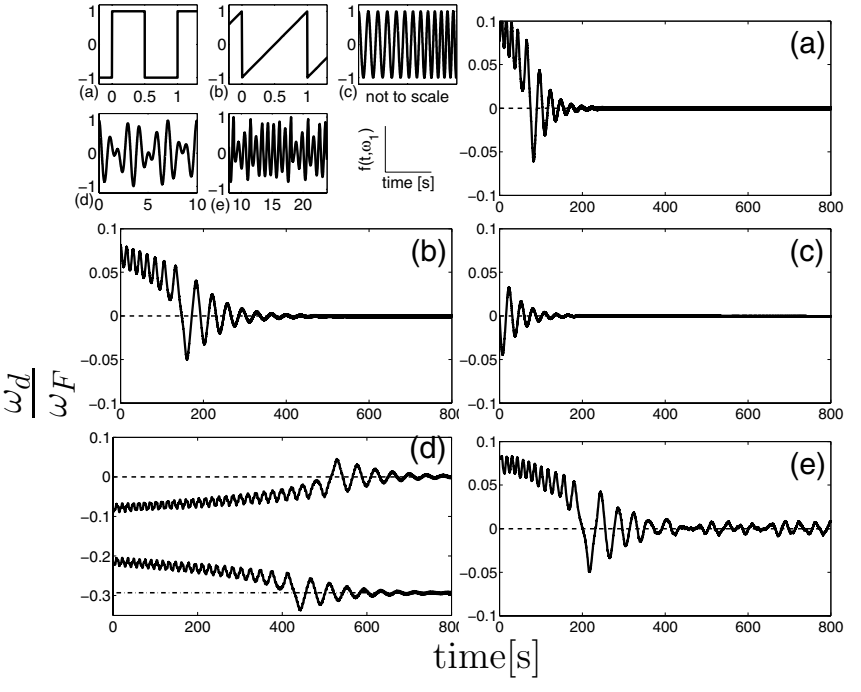


Fig. 2. On the top left panel the nonharmonic driving signals are presented. (a) Square pulse (b) Sawtooth (c) Chirp (Note that this is illustrative only since the change in frequency takes much longer as illustrated.) (d) Signal with two non-commensurate frequencies (e) Output of the Rössler system. – We depict representative results on the evolution of $\frac{\omega_d}{\omega_F}$ vs. time. The dashed line indicates the base frequency ω_F of the driving signals. In (d) we show in a representative example how the system can evolve to different frequency components of the driving signal depending on the initial condition $\omega_d(0)$.

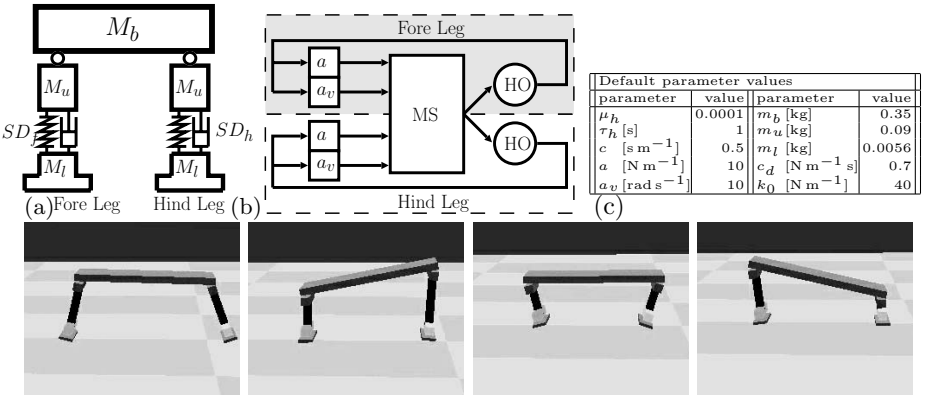
$f(t,\omega_F) = rect(\omega_F t)$, (b) Sawtooth $f(t,\omega_F) = st(\omega_F t)$, (c) Quadratic Chirp $f(t,\omega_F) = \cos(\omega_c t)$,$\omega_c = \omega_F(1 + \frac{1}{2}(\frac{t}{1000})^2)$, (d) Signal consisting of two non-commensurate frequency components, $f(t,\omega_F) = \frac{1}{2}\left[\cos(\omega_F t) + \cos(\frac{\sqrt{2}}{2}\omega_F t)\right]$, (e) Chaotic signal from the Rössler system. There are differences in the convergence speed and in the limit set. Yet, in all cases the oscillators converge to the appropriate frequencies. Very interesting cases are signals with 2 or more pronounced frequency components (such as signals (a),(b) and (d)). In this case the initial condition $\omega_d(0)$ determines to which frequency the oscillator adapts (cf Fig. 2(d)). The size of the basin of attraction is proportional to the ratio of energy of the corresponding frequency component to the total energy of the signal (due to the lack of space the data is not shown, but can be found in [18]). These simulations show that the adaptation mechanism works despite complex input signals (convergence under broad driving conditions). In the next section we will explore how these interesting properties can be applied to control a mechanical system.

3 The Adaptive Active Spring-Mass Hopper

In this section we will present the spring-mass hopper. We will first present the mechanical structure and then focus on the adaptive controller. As a general idea we will exploit the bandpass, and amplification/attenuation properties of both the mechanical system and the adaptive frequency Hopf oscillator.

Since our main interest is the adaptation in the controllers, we do not discuss the problem of mechanical stability of the locomotion. We avoid stability problems by an appropriate mechanical structure (wide feet, low center of mass), thus the feet of the robot are wide enough to ensure stability in lateral direction, i.e. the robot is essentially working in a vertical (the "sagittal") plane. The spring-mass hopper consists of 5 rigid bodies joined by rotational and linear joints (cf. Fig. 3). A prolonged cubic body is supported by two legs. The two legs are identical in their setup. A leg is made of an upper part M_u and a lower part M_l which are joined by a spring-mass system and a linear joint. The function of the linear joint is just to ensure the alignment of the body axes and is otherwise passive. The spring between the two parts of the leg is an activated spring of the form $F_f = -kd_l$, where $k = f_k(t)$ and d_l is the distance between M_u and M_l. The damper is an ideal viscous damping element of the form $F_d = -c_d v_d$, where c_d is the damping constant and $v_d = v_u - v_l$, is the relative velocity of M_u and M_l. The rotational joints between the upper part of the leg M_u and the body M_b are activated by a servo mechanism which ensures that the desired velocity v_{ref} is always maintained (cf. [2]). The choice of the spring activation function $f_k(t)$ and the choice of the desired velocity $v_{ref} = f_v(t)$ will be discussed below, when the coupling between controller and mechanical system is introduced. Due to the spring-mass property of the legs the contraction mechanism possesses a resonant frequency ω_F[1]. In other words, this type of mechanical system can be

[1] Note, that the leg can also be considered as a pendulum and thus possesses a second resonant frequency. We will not focus on this intrinsic dynamics in this article.



Default parameter values			
parameter	value	parameter	value
μ_h	0.0001	m_b [kg]	0.35
τ_h [s]	1	m_u [kg]	0.09
c [s m^{-1}]	0.5	m_l [kg]	0.0056
a [N m^{-1}]	10	c_d [N m^{-1} s]	0.7
a_v [rad s^{-1}]	10	k_0 [N m^{-1}]	40

Fig. 3. (a) The mechanical structure of the spring-mass hopper. The trunk is made up of a rigid body M_b on which two legs are attached by rotational joints. The lower part of the leg is attached by a spring-mass system SD. The lower part consist of a small rigid body. The length of the body is 0.5m (b) The coupling structure of the controller and the mechanical system used for the spring-mass hopper. The upper Hopf oscillator is used for the activation of the fore leg and the lower feedback loop for the hind leg. (c) This table presents the parameters that have been used for the simulations, unless otherwise noted. Note that this parameters can be chosen from a wide range and the results do (qualitatively) to a large extent not depend on the exact values of the parameters. Bottom row: Snapshots of the movement sequence of the spring-mass hopper when the frequency is adapted, i.e. steady state behavior (cf also movie [1]).

interpreted as a band-pass filter with the pass band around ω_F. This fact is important for the controller to be able to activate the body [4]. This will be further discussed towards the end of this section.

The *controller* of the leg consists of an adaptive frequency Hopf oscillator (Eqs. 1–3), which is perturbed by the activity of the mechanical system (see below for the exact form). The Hopf oscillator acts as a frequency selective amplifier [9], i.e. frequency components of $F_x(t)$ that are close to ω_h are amplified. Especially the setting $\mu_h = 0$ is special in the sense that the system undergoes a fundamental change at that point: For $\mu_h < 0$ the system exhibits a stable fixed point at $z = 0$, whereas for $\mu_h > 0$ a stable limit cycle occurs with radius $r = \sqrt{\mu_h}$. This phenomenon is known as a Hopf bifurcation. At $\mu_h = 0$, there is no signal oscillating at ω_h weak enough not to get amplified by the Hopf oscillator. Therefore, for that setting the Hopf oscillator can be considered an ideal amplifier. We use a setting $\mu_h \approx 0$. The *coupling* from the oscillator to the mechanics is established via the spring constants $k = f_k(t)$ and desired angular joint velocity $v_{ref} = f_v(t)$. The oscillator therefore drives both a linear actuator (the spring in the leg) and a rotational actuator (the servo in the hip). The function for the spring constant is chosen as $k = k_0 + a\frac{x_h}{r}$, where k_0 is a constant and a a coupling constant. The function for the desired velocity is chosen as $v_{ref} = a_v y_h$ where a_v is a coupling constant. The choice of this function, which introduces a $\frac{\pi}{2}$ phase lag between the spring activity and the

joint angular velocity is based on the observation of a phase lag between hip and knee joints in locomoting humans and animals. The coupling from the mechanical system to the Hopf oscillator is established via the relative velocity between upper and lower part of a limb as follows $F_x(t) = cv_d$ Fig. 3 illustrates the coupling scheme. Thus, by the coupling scheme a feedback loop is established between the oscillator and the mechanical system. If the resonant frequencies of mechanical system and the Hopf frequency match, i.e. $\omega_F \approx \omega_h$, an increase of the activity in the system is expected due to the amplifying properties of the Hopf oscillators (cf. [4]). Instead of tuning the controller manually to the appropriate frequency, we let the controller adapt its frequency. In order to achieve this adaptation, we introduce an influence of the mechanical perturbation to the evolution of ω_h via an appropriate choice of F_ω.

Adaptive Frequency The coupling of the perturbation of the mechanical system to the evolution of ω_h, allows the controller to adapt to the mechanical system and is equivalent to the perturbation arriving at the oscillator projected on the tangential direction of the limit cycle multiplied with an adaptation rate constant: $\dot{\omega}_h = -\frac{1}{\tau_h} cv_d \frac{y_h}{r} = -\frac{1}{\tau_h} F_x(t) \frac{y_h}{r}$. This is the same coupling as used for the adaptive frequency Hopf oscillator before.

4 Simulation Results

The spring-mass hopper simulation was implemented in Webots, a robot simulator with articulated-body dynamics [15]. In Table 3 we present the parameters for the spring-mass hopper that were used for the simulations unless otherwise noted. Note that this parameters can be chosen from a wide range and the qualitative results do to a large extent not depend on the exact values of the parameters.

We first show how the adaptation of the Hopf frequency ω_h leads to an excitation of the system and hopping is initiated. To avoid influence of the hip movements on the generated movement the joints are, in this case, fixed at an angle of zero degrees and $a_v = 0$, i.e. the hopper is just able to hop in place. Thus, the experiment verifies that the frequency adaptation works. As can be seen in Fig. 4, indeed, due to the adaptation the feet start to lift from the ground. In a next experiment the coupling from the oscillators to the rotational hip joints is set to its default value ($a_v = 10$). Due to the activation of the hip joints complex movements emerge. The diversity and self-organization of the movement depends on many factors, therefore in this article we will show preliminary results on the most typical locomotion pattern that was observed. This pattern resembles the bound. The movement sequence of the hopping movement is shown in a series of representative snapshots in Fig. 3. In Fig. 5, the adaptation, feet elevation and displacement of the body is presented. The average achieved velocity in steady state for $m_b = 0.35\,\mathrm{kg}$ is about $0.53\,\mathrm{ms}^{-1}$ (approx. one body length per second). The next experiment shows that the controller correctly tracks changes in the mechanical system. To demonstrate this adaptation capability the mass of the body M_b is changed from $m_b = 0.2\,\mathrm{kg}$ to $m_b = 0.4\,\mathrm{kg}$ at time $t = 40\,\mathrm{s}$. The

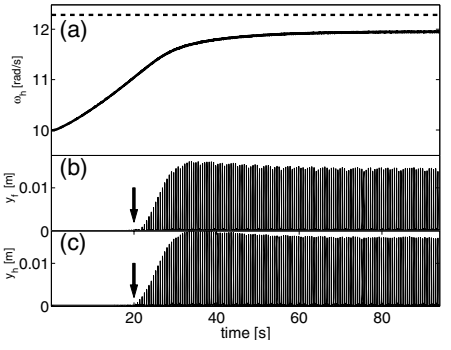


Fig. 4. Simulation results of the spring-mass hopper when the rotational joints are not activated. a) Evolution of the intrinsic frequencies of the Hopf oscillators ω_h. Note that the frequencies of both oscillators nearly coincide and therefore only one seems visible. b) Fore limb foot elevation. c) Hind limb foot elevation. The adaptation of the frequency is clearly visible and as can be seen in the feet elevation measurements the activity of the system is increased as hopping starts at around 20 s (arrow). The dashed line depicts the theoretical resonant frequency of the spring-mass system when it would not leave the ground. Due to the lift-off of the feet the real resonant frequency is smaller than the calculated value.

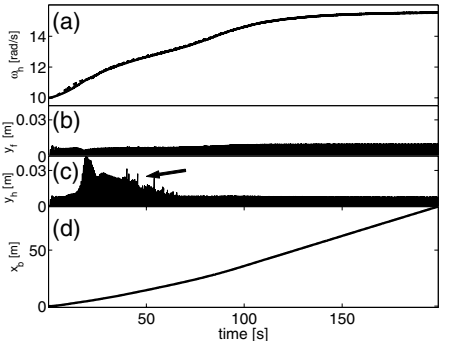


Fig. 5. Simulation results of the spring-mass hopper when the rotational joints are activated a) The frequencies of the Hopf oscillators ω_h b,c) Foot elevation $y_{f,h}$ d) Displacement of the center of mass of the body x_b. The adaptation of the frequency is clearly visible. As can be seen in the feet elevation measurements and the displacement of the body this adaptation enhances the activity of the leg and initiates a displacement of the body. Interestingly there is a burst of activation (arrow) which increases the adaptation speed before the system settles to steady state behavior.

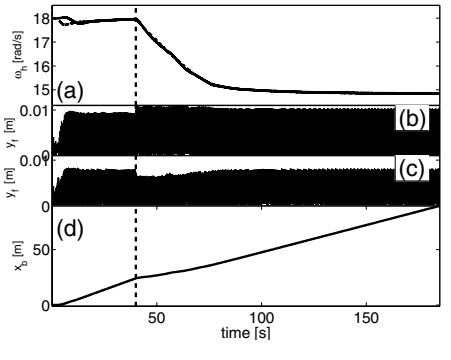


Fig. 6. Test of the adaptation capability of the controller when the mass is changed. At $t = 40$ s (dashed line) the mass of the body M_b is changed from $m_b = 0.2$ kg to 0.4 kg. See text for discussion.

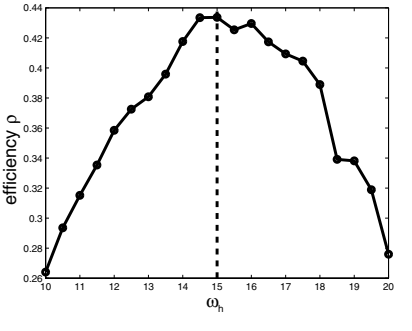


Fig. 7. Efficiency ρ vs ω_h. See text for the definition of ρ, details of measurement and experimental protocol. The dashed line indicates the frequency to which the oscillators adapt. It is clearly visible that this corresponds to the maximum of the efficiency.

results are presented in Fig. 6. As the mass is changed, the controller immediately starts to adapt and settles after about 50 s to the new resonant frequency. When looking at the displacement of the body x_b it is evident that the change in the mass slows down the system for a moment but due to the adaptation the velocity is increased again (cf also movies of the experiments [1]). The average velocity before change of mass is about $0.68\,\mathrm{ms}^{-1}$. After the change, when reached steady state behavior again, the velocity is in average $0.49\,\mathrm{ms}^{-1}$.

In a last experiment we investigate the efficiency of the hopper for forward locomotion. We define the efficiency as the ratio $\rho = \frac{\overline{v}_{x,b}}{\overline{P}_{\Sigma}}$ i.e. the ratio between average forward velocity $\overline{v}_{x,b}$ of the body and the average power $\overline{P}_{\Sigma}$ consumed by all activated joints. In order to assure steady state measurements, the learning is disabled ($\tau_h = 0$) and the experiment is repeated for different Hopf frequencies $\omega_h = [10, 10.5 \ldots, 20]$. The transient behavior is removed before the efficiency is measured. In Fig. 7, the results of the efficiency measurements are presented. The line indicates the frequency to which the system evolves if $\tau \neq 0$, thus it is clear that the adaptive frequency process finds the optimal efficiency. It is worth noting, that this optimum in efficiency does not correspond to the maximum of power consumption nor the maximum of velocity (data not shown). This is in line with the observations on animals.

5 Discussion

When introducing adaptation into a system it is important to investigate the convergence properties of the adaptation mechanism. From the mathematical point of view adaptivity on one hand and convergence and stability on the other hand are somewhat opposing requirements. We have shown, that the adaptive frequency oscillator can be driven with general, nonharmonic signals and still adapts to the frequency of the signal.

We have presented a 4-DOF spring-mass hopper with a controller based on adaptive frequency Hopf oscillators which adapts to mechanical properties of the hopper. This adaptation has the effect that the spring-mass system starts to resonate and initiates hopping locomotion, similar to the bound. The adaptation is embedded into the dynamics of the system and no pre-processing of sensory data is needed. The system shows fast adaptation to body properties.

The results presented in this paper show that with a simple control scheme, it is possible to initiate complex movements and to adapt to the body properties which are important for this movement. In order for this simple scheme to be successful it is important that the controller exploits the natural dynamics of the body. This is in line with observations in nature, where the controllers are found to be complementary to the bodies they control. In fact the adaptation can be considered a type of Hebbian learning as it maximizes the correlation between the signal perturbing the oscillator and the activity of the oscillator. Interestingly, there is no direct coupling between the controllers for the hind and the fore limb. The only coupling between the two controllers is via the mechanical structure and the environment. Nevertheless, an efficient inter-limb coordination emerges

and a fast, efficency-optimal locomotion is established. This is interesting as there is no explicit notion of velocity in the system, so it is surprising that the system optimized on velocity/power efficiency.

Immediate possible applications of such adaptive nonlinear dynamical systems are e.g. modular robotics, micro robots, robots which are difficult to model, and adaptive Central Pattern Generators (CPG) for legged locomotion. Furthermore, it will be interesting to explore the use of such adaptive systems in other fields such as in Physics, Biology and Cognitive Sciences, where oscillators are widely used model systems.

Acknowledgments. This work is funded by a Young Professorship Award to Auke Ijspeert from the Swiss National Science Foundation (A.I. & J.B.) and by the European Commission's Cognition Unit, project no. IST-2004-004370: RobotCub (L.R.). We would like to thank Bertrand Mesot for constructive comments on earlier versions of this article.

References

1. Movies of the spring-mass hopper are available at `http://birg.epfl.ch`.
2. *Open Dynamics Engine Documentation*. available at `http://ode.org`.
3. R. Blickhan and R.J. Full. Similarity in multilegged locomotion: bouncing like a monopode. *Journal of Comparative Physiology*, 173:509–517, 1993.
4. J. Buchli and A.J. Ijspeert. A simple, adaptive locmotion toy-system. In S. Schaal, A.J. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and J.A. Meyer, editors, *Proceedings SAB04*. MIT Press, Cambridge, 2004.
5. M. Buehler, R. Battaglia, R. Cocosco, G. Hawker, J. Sarkis, and K. Yamazaki. SCOUT: a simple quadruped that walks, climbs, and runs. In *Proceedings ICRA*, volume 2, pages 1707–1712, 1998.
6. J.G. Cham, J.K. Karpick, and M.R. Cutkosky. Stride period adaptation of a biomimetic running hexapod. *Int. J. of Robotics Research*, 23(2):141–153, 2004.
7. J.J. Collins and S.A. Richmond. Hard-wired central pattern generators for quadrupedal locomotion. *Biological Cybernetics*, 71(5):375–385, 1994.
8. M. de Lasa and M. Buehler. Dynamic compliant quadruped walking. In *ICRA 2001*, volume 3, pages 3153–3158, 2001.
9. V.M. Eguíluz, M. Ospeck, Y. Choe, A.J. Hudspeth, and M.O. Magnasco. Essential nonlinearities in hearing. *Phys Rev Lett*, 84(22):5232–5235, 2000.
10. Y. Fukuoka, H. Kimura, and A.H. Cohen. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. *The International Journal of Robotics Research*, 3–4:187–202, 2003.
11. R.J. Full and D.E. Koditscheck. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *J Exp Biology*, 202:3325–3332, 1999.
12. H. Geyer, A. Seyfarth, and R. Blickhan. Spring-mass running: simple approximate solution and application to gait stability. *J. Theor. Biol.*, 232(3):315–328, 2004.
13. F. Iida and R. Pfeifer. ""Cheap" rapid locomotion of a quadruped robot: Self-stabilization of bounding gait". In F. Groen et al., editor, *Intelligent Autonomous Systems 8*, pages 642–649. IOS Press, 2004.
14. H. Kimura, Y. Fukuoka, and A.V. Cohen. Biologically inspired adaptive dynamic walking of a quadruped robot. In *From Animals to Animats 8. Proceedings SAB'04*, pages 201–210. MIT Press, 2004.

15. O. Michel. Webots: Professional mobile robot simulation. *Int. J. Adv. Robotic Systems*, 1(1):39–42, 2004.
16. J. Nishii. A learning model for oscillatory networks. *Neural Networks*, 11(2):249–257, 1998.
17. M. Raibert and J.K. Hodgins. *Biol. Neural Networks in Invertebrate Neuroethology and Robotics*, chapter Legged Robots, pages 319–354. Academic Press, 1993.
18. L. Righetti, J. Buchli, and A.J. Ijspeert. Dynamic Hebbian learning for adaptive frequency oscillators. *Physica D*, 2005. submitted.
19. G. Taga. A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance. *Biol. Cybernetics*, 78:9–17, 1998.
20. K. Tsuchiya, S. Aoi, and K. Tsujita. Locomotion control of a multi-legged locomotion robot using oscillators. In *2002 IEEE Intl. Conf. SMC*, volume 4, 2002.

An Evolved Agent Performing Efficient Path Integration Based Homing and Search

R.J. Vickerstaff and E.A. Di Paolo

Centre for Computational Neuroscience and Robotics,
University of Sussex, Brighton, BN1 9QG, United Kingdom
robertvi@sussex.ac.uk
http://www.informatics.sussex.ac.uk/ccnr/people/RobertVickerstaff.html

Abstract. This paper presents analysis and follow up experiments based on previous work where a neurally controlled simulated agent was evolved to navigate using path integration (PI). Specifically, we focus on one agent, the best one produced, and investigate two interesting features. Firstly, the agent stores its current coordinates in two leaky integrators, whose leakage is partially compensated for by a normalisation mechanism. We use a comparison between four network topologies to test if this normalised leakage mechanism is adaptive for the agent. Secondly, the controller generates efficient searching behaviour in the vicinity of its final goal. We begin an analysis of the dynamical system (DS) responsible for this, starting from a simple three variable system.

1 Introduction

Path integration (PI) is a navigational method available in the absence of landmarks, and requires a compass and odometer. Information from these two sources must be continuously integrated during a journey to maintain a running estimate of the current position relative to some fixed reference point. In animal navigation studies the current estimated position is referred to as the home vector (HV), and is subject to the accumulation of error.

Several equational [1],[2],[3] and neural [4],[5],[6] models of PI exist. In our previous work [7] we presented a new neural model of PI in the desert ant *Cataglyphis fortis*, whose PI behaviour has been extensively studied [8]. We showed that it is possible, using a novel class of neural controller, to use a genetic algorithm (GA) to produce a PI system without imposing the neural mechanism for storing the HV. We also showed that using a complete model of the animal and its environment allowed the model to produce a search behaviour once the agent had returned to the vicinity of the nest. *C. fortis* also uses a searching behaviour to locate its nest, thus allowing it to home in spite of the navigation errors inherent to PI. Upon analysis we found our neural controller was very similar to Mittelstaedt's [1],[2] equational model. In this paper we present follow up work aimed at reaching a better understanding of our evolved model. We test four different network topologies to determine which one can produce the best PI system, and examine a series of equational models intermediate between Mittelstaedt's and our own.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 221–230, 2005.

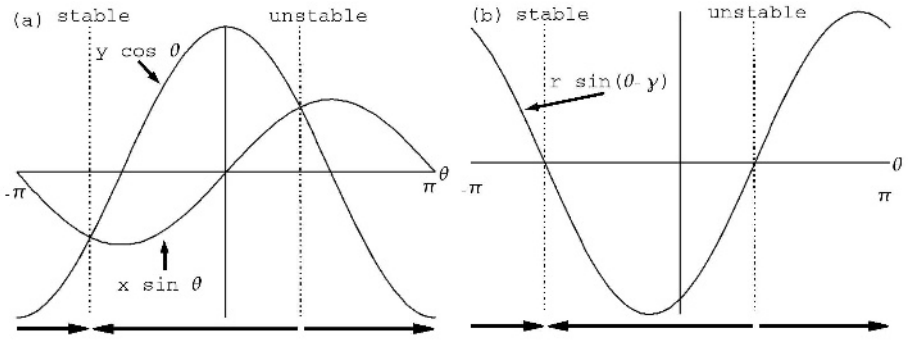


Fig. 1. Dynamics of homing using (a) equation 2 and (b) equation 3. The horizontal axis shows the animal's orientation θ, the vertical axis shows the animals rate of turn $\frac{d\theta}{dt}$ where (x, y) is the geocentric rectangular HV and (r, γ) is the geocentric polar HV. This example shows the case where (a) $x > 0, y = 2x$ and in equivalence (b) $\gamma = tan^{-1}2 = 1.107$. Both schemes lead to one stable and one unstable equilibrium heading with the same respective values.

1.1 Mittelstaedt's Bicomponent Model

PI in two dimensions can be expressed in terms of the operations required to update the HV [9], [10]. Mittelstaedt's bicomponent model [2]:

$$\frac{dx}{dt} = s\cos\theta, \frac{dy}{dt} = s\sin\theta \tag{1}$$

uses rectangular geocentric coordinates where (x, y) is the HV, s is the animal's speed and θ its compass heading. We can express the same system in polar geocentric coordinates:

$$\frac{dr}{dt} = s\cos(\theta - \gamma), \frac{d\gamma}{dt} = \frac{s}{r}\sin(\theta - \gamma)$$

where r is the animal's distance and γ its bearing from the origin. We can describe the process of homing using an equation for the animal's rate and direction of rotation as a function of the HV. Mittelstaedt's [1],[2] bicomponent model uses:

$$\frac{d\theta}{dt} = x\sin\theta - y\cos\theta \tag{2}$$

This causes the agent to always turn towards the origin $(0, 0)$. Converting this into polar form using the relations $x = r\cos\gamma, y = r\sin\gamma$ we obtain:

$$\frac{d\theta}{dt} = r\sin(\theta - \gamma) \tag{3}$$

During foraging the PI system passively updates the HV in response to the animal's movements, but when homing begins the system also directs movement,

forming a feedback system. To produce a complete model of PI navigation we therefore need to include a model of the animal's motion within its environment. Since the Mittelstaedt model updates the HV using geometrically correct formulae, in the absence of noise the HV values will also be the animal's true location. Therefore the three equations provide a simple, complete model of PI, including feedback.

2 Methods and Results

2.1 The Agent and PI Task

Our simulated agent has two compass sensors, C_L, C_R, outputting $\cos\theta, \sin\theta$ respectively (where θ is the agent's current heading), and a speed sensor, S, outputting a value between 0 and 1 indicating the agent's normalised speed. The agent's motion is controlled by an output, F, controlling the forward speed and two opposing outputs R_L, R_R controlling rotation (using $\frac{d\theta}{dt} = 150(R_L - R_R)$). To force the agent to travel at varying speeds, 70 percent noise is applied to the forward speed, and 10 percent noise to rotation. Sensor noise is 1 percent to ensure PI is feasible. To generate an initial outward excursion before homing begins the agent also has two beacon sensors B_L, B_R for phototaxis.

For each trial the agent started at the nest with a random orientation and was presented with a series of one to three randomly placed beacons which it was required to visit. Each beacon was removed when the agent reached it, and the next one activated. After the last beacon the agent's orientation was randomised, and it was held stationary for a short random time and then allowed to home using PI.

The fitness function used was such that the fittest possible agent would visit all beacons and return to the nest using direct paths at full speed, and would also search efficiently for the nest after reaching its estimate of the nest location to compensate for cumulative navigation errors. See [7] for full details.

2.2 ModCTRNN Neural Controller

Our previous work compared two types of neural controller on the PI task, the Continuous Time Recurrent Neural Network (CTRNN) [11], and a new controller the Modified CTRNN (ModCTRNN). Our results showed [7] that the ModCTRNN outperforms the CTRNN under the conditions tested, and was able to evolve a much better PI system.

A ModCTRNN network consists of ordinary CTRNN neurons, governed by the standard leaky integration equation. Weights can link from one neuron to another as normal, but can also link from a neuron to a weight. The value of a weight is variable, and performs a leaky integration of its inputs using the same form of equation as the neurons. The state equation for ModCTRNN neurons is:

$$\tau_i \frac{dv_i}{dt} = -v_i + \sum_j w_j z_j \tag{4}$$

where i indexes all neurons, j indexes all weights inputting to neuron i (if any), τ_i is a time constant, v_i is the neuron state, w_j is a weight and z_j is the activation of the sensor or neuron attached by weight j. For a neuron $z_j = \frac{1}{1+e^{-(v_j+b_j)}}$ where b_j is a bias parameter. For a sensor z_j is the current activation. The state equation for ModCTRNN weights is:

$$\alpha_i \frac{dw_i}{dt} = -w_i + \beta_i + \sum_j w_j z_j \tag{5}$$

where i indexes all network weights, j indexes all weights inputting to weight i (if any), α is a time constant, β is a bias term, w_j is a weight and z_j is the output of the neuron or sensor attached by weight j. All weights w_i are initialised to β_i, therefore any weights which receive no inputs remain constant at this value. A weight acting to modify another weight can itself be the target of modification, and so on, allowing an arbitrary degree of higher order weight change to take place.

2.3 Genetic Algorithm

An asexual GA with a population of 30 was used. Each genotype was evaluated in 10 trials per generation. Each trial the agent was tested on the PI task and assigned a fitness value. The genotype's overall fitness was the mean of 10 trials. The fittest 5 genotypes were retained unmodified in the population each generation. Each was copied 5 times to produced 25 new genotypes which were mutated and used to replace the 25 least fit genotypes. As well as mutating network parameter values, the GA also changed the number of neurons and weights present in the networks and the topology. Bilateral symmetry was imposed. Potential and weight biases were mutated within the range $[-100, 100]$. Potential and weight time constants were encoded using values between $[-2, 3]$ and mapped to their final values using $y = 10^x$, giving a range of $[0.01, 1000]$. For further details see [7].

2.4 Evolved ModCTRNN PI System

The topology of the best evolved network is shown (Fig. 2). If weights w_{L5}, w_{R5} and w_{L6}, w_{R6} are ignored, and the network can be seen to approximate Mittelstaedt's model. Weights w_{L3}, w_{R3} have a low time constant and small bias, and so they are largely at equilibrium values of $w_{L4}S, w_{R4}S$. Input to weights w_{L2}, w_{R2} are therefore $w_{R4}SC_R, w_{L4}SC_L$ respectively. Weights w_{L2}, w_{R2} have large enough time constants to remain far from equilibrium during a simulated journey, and therefore act to integrate their inputs, approximating Eqns.1 to act as the rectangular HV. Eqn.2 is implemented since the inputs to R_L, R_R are $w_{L2}C_L, w_{R2}C_R$ and since the two output neurons act in opposition to cause rotation.

Here we take two differing methodologies to analyse the PI system. Firstly, we investigate the effect of network topology on fitness, by constructing four

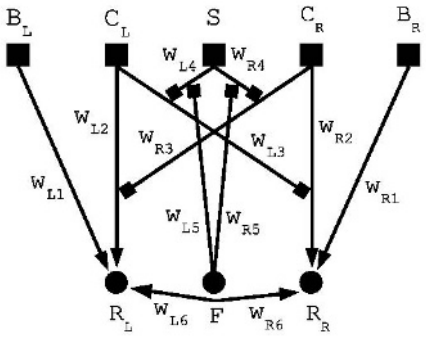


Fig. 2. The original ModCTRNN PI network. $B_{L/R}$, left/right beacon sensor, $C_{L/R}$, left/right compass sensor, $R_{L/R}$, left/right rotation motor neuron, S, speed sensor and F, forward motor neuron. Arrows are weights. Lines ending in small squares are weights which modify other weights. $w_{L1/R1} = 12.0720$, $w_{L2/R2} : \alpha = 8.4355, \beta = 0.0001$, $w_{L3/R3} : \alpha = 0.0123, \beta = 2.0477$, $w_{L4/R4} : \alpha = 5.1753, \beta = -98.7613$, $w_{L5/R5} = 65.9304$, $w_{L6/R6} = -3.5159$, $F : \tau = 0.0489, b = 42.8689$, $R_{L/R} : \tau = 0.0106, b = 0.2994$.

network topologies derived from the original, and evolving multiple populations for each. Secondly, we construct simplified DS models of the original in a noiseless, accurate numerical integration simulator and try to classify the types of behaviour we observe.

2.5 Evolving with Four Network Topology Classes

Four network topologies were constructed: *class 1* the original network topology, *class 2* the original minus weights $w_{L5/R5}$, *class 3* the original minus weights $w_{L6/R6}$ and *class 4* the original minus weights $w_{L5/R5}$ and weights $w_{L6/R6}$. Thus class 4 contains only what is neccessary to implement Mittelstaedt's model, and classes 2 and 3 are the two possible intermediate topologies between it and the evolved network. These were used to re-evolve PI behaviour starting from weight values of zero in GA runs where the network topology was not allowed to mutate. Eight runs of 1500 generations were performed for each. The best agent in each of the final populations was tested in 1000 trials and scored for the number of times it reached the nest within the time limit. The best agent in each class was then evolved for a further 20000 generations, and tested against the agent from 1500 generations. Only the class 4 agent showed an improvement, and was used in place of the generation 1500 agent. The generation 1500 agent was retained in the other classes. The four agents were then ranked for fitness. Each was tested six times for 1000 trials against the agents immediately above and below it, to determine if the ranking was statistically significant. The ranking order is class 1 > class 2 > class 3 > class 4. All differences were significant at $p < 0.01$ (Mann

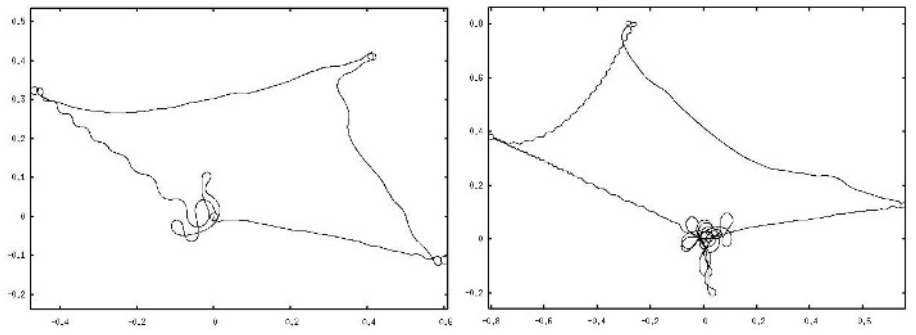


Fig. 3. Behaviour of the full evolved model. Left: the agent visits three beacons, starting from the nest (bottom centre) travelling anticlockwise. The beginning of a search pattern is visible before it reaches the nest. Right: a similar anticlockwise journey, but here the nest has been removed so that the agent searches until the trial times out.

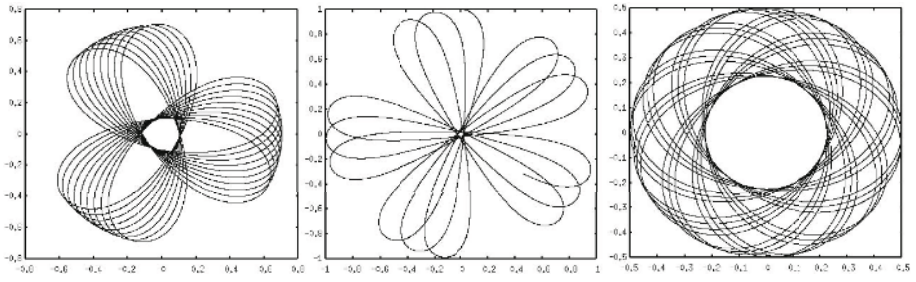


Fig. 4. Agent trajectories plotted in x, y space using the basic Mittelstaedt model with maximum turning rate parameter $k = 8$, initial conditions $x_0 = 0, \theta_0 = 0$ and, moving from left to right $y_0 = 0.1, y_0 = 0.5, y_0 = 1.0$. Clearly the initial condition influences the pattern.

Whitney U test). Percentage success rates for classes 1 to 4 were 97.4, 89.1, 81.1 and 69.5 respectively over 6000 trials.

2.6 Analysis of Searching Behaviour

The ModCTRNN PI agent shows a searching behaviour when it reaches its estimate of the nest location (Fig.3). The shape of this trajectory is important in determining how likely the agent is to locate its nest within the time limit. Trajectories whose density profile closely matches that of the probable nest location (which is usually not exact where the agent expects it to be due to cumulative PI errors) should be most efficient.

In the absence of noise Mittelstaedt's model can constitute a complete PI system, since the HV is also the agent's true location. We begin with this system, assuming the agent's speed is constant at 1, and build in further features derived

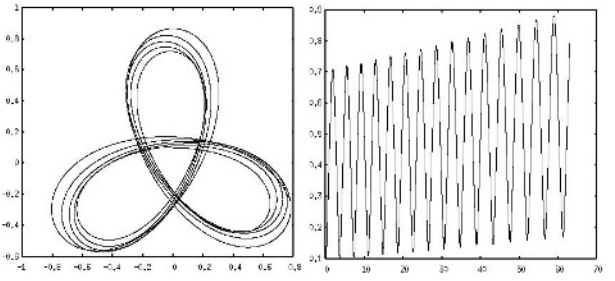


Fig. 5. The basic Mittelstaedt model ($k = 8, x_0 = 0, y_0 = 0.01, \theta_0 = 0$) augmented with a decay term in the turning rate equation, $e^{-\alpha t}$, where $\alpha = 0.01$. Left, the trajectory plotted in x, y space, right, the agent's distance from the origin plotted over time. The pattern clearly gets wider over time.

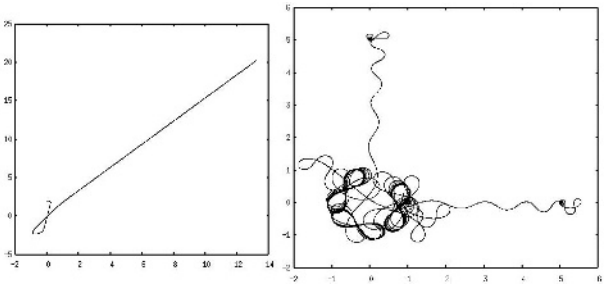


Fig. 6. Left: the addition of sigmoidal functions to the turning rate equation of the basic model ($k = 8, x_0 = 0, y_0 = 2, \theta_0 = 0$) allows the agent to become trapped on an outward diagonal trajectory. Right: the addition of stateful output units ($\tau = 1$) to the basic model ($k = 8, \theta_0 = 0$) causes the trajectories to differentiate into an initial homing phase, followed by a searching phase whose size is independent of the initial release point for the three cases tested here ($x_0 = 0, y_0 = 5$ and $x_0 = 5, y_0 = 0$ and $x_0 = 1, y_0 = 0$).

from the evolved network, in order to build intuition about how the search patterns are generated. We remove all noise from the system to aid analysis, but note that in reality noise would cause uncertainty in the nest location, and could knock the agent out of unstable trajectories. The three state equations to be used are:

$$\frac{dx}{dt} = \cos\theta, \frac{dy}{dt} = \sin\theta, \frac{d\theta}{dt} = k(x\sin\theta - y\cos\theta)$$

where k is the agent's maximum turning rate. Efficiency can obviously increase the faster an agent travels since it covers more ground, but we are interested in the efficiency of the *shape* of the search pattern. Therefore we transfer a degree of freedom from the agent's speed to its turning rate. In polar form:

$$\frac{dr}{dt} = \cos(\theta - \gamma), \frac{d\gamma}{dt} = \frac{1}{r}\sin(\theta - \gamma), \frac{d\theta}{dt} = kr\sin(\theta - \gamma)$$

This system can be derived from the evolved network by assuming the HV is updated without errors, that the output neurons have a linear response and are always at equilibrium and by neglecting w_{L6}, w_{R6}. The rectangular and polar systems were numerically integrated using the Runge-Kutta fourth order method [12] using a step size of 0.00005, and plotted together to check they were identical. The system displays behaviour similar to that of a spirograph [13] (see Fig.4). Trajectories show no division into homing and search phases, unlike the evolved agent. Defining $\delta = \theta - \gamma$, and plotting in (r, δ) phase space (not shown) all initial conditions (except those with $\delta_0 = \pm\pi$ or 0) appear to give closed trajectories in (r, δ) space, with no initial transient. Two fixed points are at $r = \sqrt{\frac{1}{k}}, \delta = \pm\frac{\pi}{2}$, which linearisation shows are neutrally stable centres, corresponding to circles in (r, γ) space. Defining $\phi = \delta \pm \frac{\pi}{2}$ and $\rho = r - \sqrt{\frac{1}{k}}$ we have a reversible system, since $\dot{\rho}(\rho, -\phi) = -\dot{\rho}(\rho, \phi)$ and $\dot{\phi}(\rho, -\phi) = \dot{\phi}(\rho, \phi)$ (where $\dot{\rho}, \dot{\phi}$ indicate derivatives) [14]. It follows that all trajectories sufficiently close to the fixed points are closed, implying that all loops of a given trajectory in (r, γ) space are congruent in this region.

As stated in our previous work [7], the time constants of weights w_{L2}, w_{R2} are small enough to allow the HV to decay significantly during the course of a journey, but this decay is approximately compensated for by the decay of weights w_{L4}, w_{R4}. We refer to this as leakage normalisation, since it scales down the HV integrator inputs to approximately balance the decay of the HV values, thus restoring accurate PI. If we assume this process is perfect we can derive an extension of the above DS which includes the effect of the HV decay process. The HV is still accurate, but is scaled by an exponentially decaying coefficient over the course of the journey. This can be modelled by amending $\frac{d\theta}{dt}$:

$$\frac{d\theta}{dt} = ke^{-\alpha t}(x\sin\theta - y\cos\theta)$$

$\frac{d\theta}{dt}$ will eventually reach zero in this scheme, (which does not happen in the full network model), and the agent will stop turning, but it is sufficient to show that the leakage normalisation mechanism can generate a search pattern which gets wider over time (see Fig.5), as is seen in *C. fortis* [15].

The output units of the ModCTRNN model are sigmoidal, but do not reach saturation during homing and search (data not shown). To see why output saturation might have a deleterious effect on homing and search we can amend $\frac{d\theta}{dt}$:

$$\frac{d\theta}{dt} = k\left(\sigma(x\sin\theta) - \sigma(y\cos\theta)\right)$$

where $\sigma(x) = \frac{1}{1+e^{\ x}}$. Saturation causes dead zones to appear around the two equilibrium values of θ (the stable homeward and unstable outward directions), where both outputs are saturated on or off, when x and y have similar magnitudes, thus making the unstable equilibrium neurally stable. This can trap the

agent on an outward trajectory along a diagonal (see Fig.6). Thus we can suggest a good reason why the outputs of R_L, R_R should be unsaturated during homing and search.

The output units in the ModCTRNN are stateful leaky integrators. We have so far treated them as being at equilibrium, but we can include stateful linear output units in the DS model as follows:

$$\frac{d\theta}{dt} = k(R_L - R_R), \frac{dR_L}{dt} = \frac{1}{\tau}(-R_L + x\sin\theta), \frac{dR_R}{dt} = \frac{1}{\tau}(-R_R + y\cos\theta)$$

Now for the first time, we obtain trajectories with distinct homing and search phases (see Fig.6). The example in the figure shows the agent converging onto a figure of eight shape from three initial positions. The figure eight is the same size irrespective of the initial release point. The agent is also seen to oscillate either side of the direct homeward trajectory during homing as the full model does.

3 Discussion

The topology experiment results strongly suggest that the weights in the evolved network which cannot be considered as directly implementing Mittelstaedt's model (weights $w_{L5}, w_{R5}, w_{L6}, w_{R6}$) none the less perform some adaptive role, and that removing w_{L6}, w_{R6} has a greater effect than removing w_{L5}, w_{R5}. Our DS models suggest that leakage normalisation could cause the agent's search pattern to get wider over time, which might increase the search efficiency, but our previous work [7] has only shown weak evidence for this, since individual searches do not get noticeably wider as w_{L4}, w_{R4} decays. We therefore still cannot state the function of the normalisation process. Consideration of linear stateful output neurons shows the agent's trajectory already significantly different from that of Mittelstaedt's model, and considerably more complex (now with 5 variables, not 3). Since w_{L6}, w_{R6} cannot affect HV update, they must function to further modify the agent's search pattern towards that seen in the full model. Further work along similar lines will be needed to finally resolve these issues.

Overall, we conclude that the use of the ModCTRNN model has proved useful for evolving a PI agent, and that whilst the model is slightly more complex than the CTRNN, the resulting PI network is both simple and amenable to considerable analysis and understanding. We hope the ModCTRNN will prove useful for other evolutionary robotics projects in the future.

Acknowledgements

This work was financially supported by BBSRC grant number 02/A1/S/08410. We wish to thank Thomas Collett, Kyran Dale and Paul Graham for their many helpful suggestions during the preparation of this work.

References

1. Mittelstaedt, H.: Control systems of orientation in insects. Annu. Rev. Entomol. **7** (1962) 177–198
2. Mittelstaedt, H.: Analytical cybernetics of spider navigation. In Neurobiology of arachnids (ed. F. Barth) (1985) 298–318 Berlin:Springer.
3. Müller, M. and Wehner, R.: Path integration in desert ants, *Cataglyphis fortis*. Proc. Natl. Acad. Sci. USA **85** (1988) 5287–5290
4. Hartmann, G. and Wehner, R.: The ant's path integration system: a neural architecture. Biol. Cybern. **73** (1995) 483–497
5. Wittmann, T. and Schwegler, H.: Path integration - a neural model. Biol. Cybern. **73** (1995) 569–575
6. DaeEun, K. and Hallam, J.: Neural network approach to path integration for homing navigation. Proceedings of Simulation of Adaptive Behaviour: From Animals to Animats 6. MIT Press (2000)
7. Vickerstaff, R. J. and Di Paolo, E. A: Evolving neural models of path integration. (submitted)
8. Wehner, R.: Desert ant navigation: how miniature brains solve complex task. J. Comp. Physiol. A. **189** 579–588
9. Benhamou, S. and Séguinot, V.: How to find one's way in the labyrinth of path integration models. J. Theor. Biol. **174** (1995) 463–466
10. Maurer, R. and Séguinot, V.: What is modelling for? A critical review of models of path integration. J. Theor. Biol. **175** (1995) 457–475
11. Beer, R. D.: Intelligence as adaptive behaviour: an experiment in computational neuroethology. Academic Press: Boston. (1990)
12. Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P.: Numerical recipes in C: 2nd edition Cambridge University Press
13. Weisstein, E. W.: Spirograph. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Spirograph.html
14. Strogatz, S. H.: Nonlinear dynamics and chaos. Perseus Group Books (2001)
15. Wehner, R. and Srinivasan, M. V.: Searching behaviour of desert ants, genus *Cataglyphis* (Formicidae, Hymenoptera). J. Comp. Physiol. A. **142** (1981) 315–338

Evolving Neural Mechanisms for an Iterated Discrimination Task: A Robot Based Model

Elio Tuci, Christos Ampatzis, and Marco Dorigo

IRIDIA, Université Libre de Bruxelles - Bruxelles - Belgium
{etuci, campatzi, mdorigo}@ulb.ac.be

Abstract. This paper is about the design of an artificial neural network to control an autonomous robot that is required to iteratively solve a discrimination task based on time-dependent structures. The "decision making" aspect demands the robot "to decide", during a sequence of trials, whether or not the type of environment it encounters allows it to reach a light bulb located at the centre of a simulated world. Contrary to other similar studies, in this work the robot employs environmental structures to iteratively make its choice, without previous experience disrupting the functionality of its decision-making mechanisms.

1 Introduction

Evolutionary Robotics (ER) is a methodological tool for the design of robots' controllers. Owing to its properties, ER can also be employed to study the evolution of behaviour and cognition from a perspective complementary to classic biological/psychological methods (see [4]).

Given the current "status" of their research field, ER practitioners focus not only on studies with an explicit bearing on engineering or biological literature but also on studies which aim to further develop their methods. For example, several research works have focused on the modelling and exploitation of alternative controllers for autonomous robots—e.g., spiking networks [7], and gas networks [5]. In general, these works look at how to exploit evolution to shape these controllers rather than at the complexity and the significance of the evolved behaviour. Contrary to these, other works are more focused on the evolution of novel—i.e., never evolved yet—and complex behaviour. For example, some works exploited "classic" neural structures to evolve controllers for agents capable of non-reactive or learning behaviour [8]. The results of these studies should be considered as a "proof-of-concept": they show that the type of control structure employed can be shaped by evolutionary algorithms to provide the robot with the underlying mechanisms required to solve the task at hand.

The work illustrated in this paper belongs to the latter category. To the best of our knowledge, this is the first study in which a single (i.e., non modularised) dynamic neural network has been shaped to control the behaviour of an autonomous robot engaged in an iterated discrimination task.[1]. The task requires

[1] A literature review of the field can be found in [9].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 231–240, 2005.

navigation within a circular arena in order to approach a light bulb located at the centre of this simulated world. The "decision making" aspect requires the robot "to iteratively decide", during a sequence of trials, whether or not the types of environment it encounters allow it to accomplish its task. The difficulty of the task lies, on the one hand, in the nature of the discrimination problem, which requires the integration of sensory information over time; on the other hand, in the design of decision making mechanisms to carry out the iterated discrimination task. That is, this task requires the robot to possess memory structures which do not lose their functionality due to potentially disruptive effects of the previous experience—i.e., the nature and the amount of discriminations already made. The results show that dynamic neural networks can be successfully designed by evolution to allow a robot to iteratively solve the discrimination task based on time-dependent cues. We also provide an analysis which gives some hints on the strategy employed by the best evolved robot to solve the task.

2 Methods

Description of the task. At the start of each trial, the simulated robot is placed in a circular arena with a radius of 110 cm (see Fig. 1). The arena is simulated as a toroidal world; that is, if the robot traverses the world's boundaries from one side, it comes in from the other side at the anti-diametrical position. At the centre of this world there is a light bulb that is always turned on during a trial. The light can be perceived up to a distance of 90 cm. Between 90 cm and 110 cm of distance to the bulb, the robot does not perceive any light. We refer to this area of the arena as the *dark* zone. The robot perceives the light through its ambient light sensors. The colour of the arena floor is white except for a circular band, centred around the lamp, within which the floor is in shades of grey. The circular band covers an area between 40 cm and 60 cm from the light. The band is divided in three sub-zones of equal width but coloured differently—i.e., light

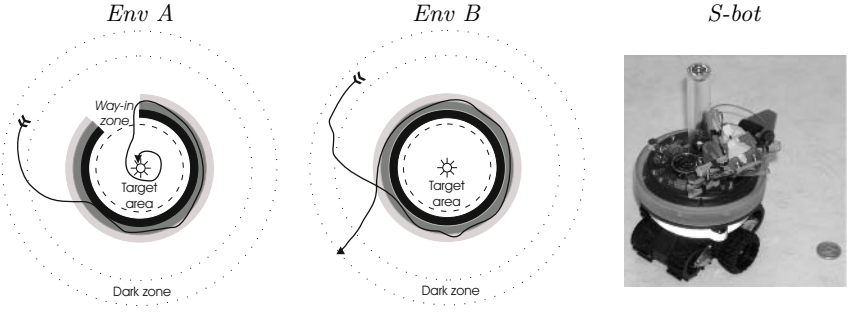


Fig. 1. Depiction of the task and picture of a *s-bot* close to a 1 Euro coin. The *dark* zone is the area within the dotted circles. The target area, centred on the light, is indicated by the dashed circle. The continuous arrows are examples of good navigational strategies.

grey, dark grey, and black. The robot perceives the colour of the floor through its floor sensor, positioned under its chassis.

The robot can freely move within the arena as well as on the circular band, but it is not allowed to cross the black edge of the band close to the light. This edge can be imagined as an obstacle or a trough, that prevents the robot from further approaching the light. Whenever the robot crosses the black edge, the trial is unsuccessfully terminated. The light grey and the dark grey zones are meant to work as a warning signal which indicates to the robot how close it is to the danger—i.e., the black edge. There are two types of environment. In one type—referred to as *Env A*—the band presents a discontinuity (see Fig. 1, left). This discontinuity, referred to as the *way in* zone, is a sector of the band in which the floor is white. In the other type—referred to as *Env B*—the band completely surrounds the light (see Fig. 1, middle). The *way in* zone represents the path along which the robot is allowed to safely reach the light in *Env A*. The robot cannot reach the proximity of the light in *Env B*.

At the start of each trial, the robot does not know in which type of environment it is located. It finds itself positioned in the *dark* zone with a random orientation. At this time its task is to explore the arena, in order to get as close as possible to the light. If it encounters the circular band it has to start looking for the *way in* zone in order to continue approaching the light. If it finds the *way in* zone, the robot has to get closer to the light and remain in its proximity for 10s. After this time, the trial is successfully terminated and the robot is randomly re-positioned in the *dark* zone. If there is no *way in* zone (i.e., the current environment is *Env B*), the robot should be capable of (a) "recognising" the absence of the *way in* zone, (b) notifying by a sound signal the absence of the *way in* zone, (c) coming back to the *dark* zone by performing anti-phototaxis. Back in the *dark* zone either because re-positioned or because returned there, the robot has to "prepare" itself for a new trial in which the characteristics of the environment are unknown. The transition between two consecutive trials is particularly complex in case the robot has just concluded a trial in *Env B*. This transition requires the robot to turn the sound off and to switch from anti-phototaxis (i.e., the last behaviour performed in *Env B*) to random walk and then phototaxis once the light falls again within its perceptual field.

This task is very similar to the one described in [9] since the robot is required to make use of a temporal cue in order to discriminate between *Env A* and *Env B*. This discrimination is based on the persistence of the perception of a particular sensorial state (e.g., the perception of the grey floor, the light, or both) for the amount of time that, given the trajectory and speed of the robot, corresponds to the time required to make a loop around the light. In other words, if the perception of a particular sensorial state common to both types of environment lasts significantly long with respect to the speed and trajectory of the robot, then that sensorial state might be used by the robot to "conclude" that there is no *way in* zone, and a tone has to be emitted (see [9] for more details).

However, with respect to [9], this task is meant to be a step further in the evolution of decision making mechanisms based on time-dependent structures.

In [9], we studied the evolution of decision making mechanisms for a one shot discrimination task by simply resetting the robot's controller (i.e., set to 0 the cell potential of the neurons) at the beginning of each trial. The resetting "facilitates" the task of discriminating between *Env A* and *Env B* since (a) the integration of the sensorial state which eventually leads to the emission of the sound signal is not disrupted by the type and the amount of previous experience; (b) the robot does not need to terminate the emission of the sound signal, since, given the way in which sound is implemented, such an event is automatically determined by the resetting; (c) the robot does not need to "recognise" the end of the current trial and the beginning of a new one, since such transition implies the resetting of the activation values of the neurons of its controller. In other words, each trial is for the robot like a new life in which, starting from the same internal state, a single decision has to be made.

The task described in this paper is made significantly more complex with respect to what shown in [9] by (a) avoiding to impose the resetting of the robot controller at the beginning of each trial, and consequently by (b) letting the robot autonomously develop the conditions which set the end of a trial and the beginning of a new one. If the robot controller is not reset at the beginning of a trial, the decision to be made in the trials following the first one, will necessarily be carried out by mechanisms which have already been "shaped" by previous experience.[2] Therefore, it is important that the functionality of the decision making mechanisms employed by the robot are not disrupted by previous experience. In other words, discriminating between *Env A* and *Env B* requires the robot to make use of memory structures to integrate over time a particular sensorial state. Carrying out such an iterated discrimination task requires the robot to possess memory structures which do not lose their functionality due to potential disruptive effects of the previous experience—i.e., the nature and the amount of discriminations already made. Furthermore, the robot should be able to exploit its perception in order to establish when a trial ends and a new one starts. This is particularly important at the end of an exploration in *Env B*, in which the robot should conclude the trial by emitting a tone and moving away from the light and should begin the new trial with the sound turned off and performing light seeking behaviour. These changes (i.e., sound on - sound off, anti-phototaxis - phototaxis) have to be triggered by perceptual states which ideally set the end of a trial and the start of a new one.

Several implementation details such as (i) requiring the robot to perform anti-phototaxis in *Env B*, (ii) the introduction of the *dark* zone, and (iii) the toroidal world, have been introduced to make sure that the robot's sensory experience can potentially provide the support the robot needs in order to make iterated choices. For example, a robot that successfully terminates a trial in *Env B* can exploit the perceptual states associated with performing anti-phototaxis and with its presence in the *dark* zone to "prepare" itself for the new trial (a) by turning the sound signalling off, and (b) by adjusting its internal state so that it will be ready for a new discrimination task. In particular, being repositioned in

[2] In our model, it is the neuron's cell potential to be modified by the robot's experience.

the *dark* zone after a success in *Env A*, or reaching the *dark* zone after a success in *Env B*, are two events that can be unambiguously employed by the robot in order to establish the end of the current trial and the beginning of the following one. In the absence of a global framework of reference (e.g., a compass), the toroidal world makes it easier for a robot to navigate in the *dark* zone in order to reach the area in which the light source can be perceived.

The simulation model. The robot and its world are simulated using the "minimal simulation" technique described in [6]. This technique uses high levels of noise to guarantee that the simulated controller transfers to the physical robot with no loss of performance (see [1]). Our simulation models some of the hardware characteristics of the real *s-bots*. The *s-bots* are small wheeled cylindrical robots, 5.8 cm of radius, equipped with infrared proximity sensors, light and humidity sensors, accelerometers, and omni-directional camera (see Fig. 1, right, and also `http://www.swarm-bots.org/` for more details). In particular, our simulated *s-bot* is provided with four ambient light sensors, placed at $-112.5°$ (A_1), $-67.5°$ (A_2), $67.5°$ (A_3), and $112.5°$ (A_4) with respect to its heading, a floor sensor positioned facing downward on the underside of the robot (F), an omni-directional sound sensor (S), and a loud-speaker. The motion of the robot is implemented by the two wheel actuators. Light levels change as a function of the robot's distance from the lamp. Light sensor activation values are taken from a look-up table which contains sampled information from the real robot. The ground sensor detects the level of grey of the floor. The robot floor sensor outputs the following values: 0 if the robot is positioned over the white floor; $\frac{1}{3}$ if the robot is positioned over the light grey floor; $\frac{2}{3}$ if the robot is positioned over the dark grey floor; 1 if the robot is positioned over the black floor. The simulated speaker produces a binary output (on/off); the sound sensor has no directionality and intensity features. Concerning the function that updates the position of the robot within the environment, we employed the Differential Drive Kinematics equations, as presented in [2]. 10% uniform noise was added to the light sensor readings, the motor outputs and the position of the robot.

The controller and the evolutionary algorithm. Fully connected, eight neuron continuous time recurrent neural networks (CTRNNs) are used. All neurons are governed by the following state equation:

$$\frac{dy_i}{dt} = \frac{1}{\tau_i}\left(-y_i + \sum_{j=1}^{8} \omega_{ji}\sigma(y_j + \beta_j) + gI_i\right) \qquad \sigma(x) = \frac{1}{1+e^{-x}} \tag{1}$$

where, using terms derived from an analogy with real neurons, y_i represents the cell potential, τ_i the decay constant, β_j the bias term, $\sigma(y_j + \beta_j)$ the firing rate, ω_{ji} the strength of the synaptic connection from neuron j^{th} to neuron i^{th}, I_i the intensity of the sensory perturbation on sensory neuron i. Four neurons receive input (I_i) from the robot sensors: neuron N_1 takes input from $\frac{A_1+A_2}{2}$, N_2 from $\frac{A_3+A_4}{2}$, N_3 from F, and N_4 from S. Neurons N_1, N_2, and N_3 receive as input a

real value in the range [0,1], while neuron N_4 receives a binary input (i.e., 1 if a tone is emitted, otherwise 0). The other neurons do not receive any input from the robot's sensors. The cell potential (y_i) of the 6^{th} neuron, mapped into [0,1] by a sigmoid function (σ), is used by the robot to control the sound signalling system (i.e., the robot emits a sound if $y_6 >= 0.5$). The cell potentials (y_i) of the 7^{th} and the 8^{th} neuron, mapped into [0,1] by a sigmoid function (σ) and then linearly scaled into $[-6.5, 6.5]$, set the robot motors output. The strength of synaptic connections ω_{ji}, the decay constant τ_i, the bias term β_j, and the gain factor g are genetically encoded parameters. Cell potentials are set to 0 any time the network is initialised or reset, and circuits are integrated using the forward Euler method with an integration step-size of 0.1.

A simple generational genetic algorithm is employed to set the parameters of the networks [3]. The population contains 100 genotypes. Generations following the first one are produced by a combination of selection with elitism, recombination and mutation. For each new generation, the three highest scoring individuals ("the elite") from the previous generation are retained unchanged. The remainder of the new population is generated by fitness-proportional selection from the 70 best individuals of the old population. Each genotype is a vector comprising 81 real values (64 connections, 8 decay constants, 8 bias terms, and a gain factor). Initially, a random population of vectors is generated by initialising each component of each genotype to values chosen uniformly random from the range [0,1]. New genotypes, except "the elite", are produced by applying recombination with a probability of 0.3 and mutation. Mutation entails that a random Gaussian offset is applied to each real-valued vector component encoded in the genotype, with a probability of 0.13. The mean of the Gaussian is 0, and its standard deviation is 0.1. During evolution, all vector component values are constrained to remain within the range [0,1]. Genotype parameters are linearly mapped to produce CTRNN parameters with the following ranges: biases $\beta_j \in$ [-2,2], weights $\omega_{ji} \in [-6, 6]$ and gain factor $g \in$ [1,12]. Decay constants are firstly linearly mapped onto the range $[-0.7, 1.7]$ and then exponentially mapped into $\tau_i \in [10^{-0.7}, 10^{1.7}]$. The lower bound of τ_i corresponds to a value slightly smaller than the integration step-size used to update the controller; the upper bound corresponds to a value slightly bigger than the average time required by a robot to reach and to perform a complete loop of the band in shades of grey.

The evaluation function. During evolution, each genotype is coded into a robot controller, and is evaluated ten times, 5 trials in *Env A*, and 5 trials in *Env B*. The sequence of environments within the 10 trials is chosen randomly. Each trial (e) differs from the others in the initialisation of the random number generator, which influences the robot starting position and orientation, the position and amplitude of the *way in* zone, and the noise added to motors and sensors. The width of the *way in* zone can vary from 45° to 81°. Within a trial, the robot life-span is 80 s (800 simulation cycles). In each trial, the robot is rewarded by an evaluation function f_e which corresponds to the sum of the following four components:

1) C_1 rewards fast movement to the target area. $C_1 = \frac{d_i - d_c}{d_i}$ where d_i and d_c represent respectively the initial and the current Euclidean distance between the robot and the light bulb. In *Env A*, C_1 is set to 1 if the robot terminates the trial less than 35 cm away from the light bulb. In *Env B*, C_1 is set to 1 as soon as the robot reaches the circular band without crossing the black edge in the direction of the light.
2) C_2 rewards movements away from the light. $C_2 = \frac{d_c}{d_{max}}$ if trial in *Env B*, 0 if trial in *Env A* or if $C_1 < 1$ ($d_{max} = 110$ cm).
3) C_3 rewards agents that never signal in *Env A* and that always signal in *Env B*. C_3 is set to 1 if the robot signals properly, 0 otherwise. The robot is considered to have signalled only if it has done so being closer than 70 cm from the light. By doing so, we create an area between 70 cm and 110 cm from the light that the robot can use to turn the sound off at the end of a trial in *Env B*.
4) C_4 rewards movements toward the light. $C_4 = 1 - \frac{k}{T}$ if trial in *Env A*, 0 otherwise. k is the number of simulated time-steps the robot spent to reach the target area, and $T = 800$ is the total number of simulated time-steps available to the robot. An important feature of this evaluation function is that it simply rewards agents that make a proper use of their sound signalling system, without directly interfering with the nature of the discrimination strategies.

3 Results

Ten evolutionary simulations, each using a different random initialisation, were run for 7000 generations. We examined the best individual of the final generation from each of these runs in order to establish whether they evolved the required behaviour. Recall that the robot is successful in *Env A* if it reaches the target area without emitting any sound signal; it is successful in *Env B* if (a) it reaches the circular band, (b) signals the absence of the *way in* zone by emitting a tone, and (c) comes back to the *dark* zone (anti-phototaxis).

During the post-evaluation phase, each of the ten best evolved controllers was subjected to a set of 252 different re-evaluations. Since a re-evaluation is composed of 10 trials, out of which 5 are *Env A* and 5 are *Env B*, 252 ($\frac{10!}{5!\cdot 5!}$) is the number of possible evaluations which differ in the order of presentation of *Env A* and *Env B*. 2520 is the total number of trials experienced by each robot during the post-evaluation, half of which in *Env A* and half in *Env B*. Note that during evolution, each robot experienced only a particular sequence of 5 trials in *Env A* and 5 trials in *Env B*. Since the robot controller is reset only at the beginning of each evaluation, the order of presentation of the types of environment might bear upon its performance. A robot that results successful in the post-evaluation is one which employs a strategy which is effective regardless the sequence of environments.

During the post-evaluation phase, we looked at the robot's capability to reach the light bulb (*Succ.*) in *Env A*, without making any error. Errors can be of three types: error (I) refers to the case in which the robot emits a sound signal, error (II) refers to the case in which the robot crosses the black edge of the band, error

Table 1. Results of post-evaluation showing the performance of the best evolved controllers of each run. The percentage of success (*Succ.*) and the percentage of errors (I, II, III in *Env A*, and IV, V, VI, VII in *Env B*) over 252 evaluations are shown for both *Env A* and *Env B*. Additionally, the average offset Δ, its standard deviation (degrees), and the number of successful trials (*n.*) are shown for *Env B*.

run	*Env A*				*Env B*							
	Succ.	*Types of Error (%)*			*Succ.*	*Types of Error (%)*				offset Δ		
		I	II	III		IV	V	VI	VII	*Avg.*	*Std*	*n.*
n. 1	83.17	7.77	4.76	0.15	1.42	14.92	0.00	4.12	79.52	21.25	115.86	18
n. 2	94.12	5.87	0.00	0.00	96.42	1.34	0.00	0.00	2.22	50.52	124.08	1215
n. 3	22.85	0.00	77.06	0.0	0.0	0.23	0.0	99.76	0.0	—	—	0.0
n. 4	84.92	14.68	0.23	0.00	98.57	0.0	0.0	0.0	1.42	-81.12	39.32	1242
n. 5	86.19	8.33	5.23	0.07	81.58	0.07	0.15	15.23	2.93	-10.8	79.05	1028
n. 6	33.17	10.07	50.55	6.19	92.93	0.00	0.00	0.079	6.98	6.1	104.75	1171
n. 7	88.49	11.11	0.39	0.00	97.53	0.00	0.00	0.00	2.46	-15.81	80.87	1229
n. 8	82.93	16.50	0.47	0.07	96.19	1.11	0.00	0.00	2.69	-60.04	82.16	1212
n. 9	99.68	0.07	0.23	0.00	95.87	2.06	0.00	0.23	1.82	71.03	50.89	1208
n. 10	59.04	40.95	0.00	0.00	98.57	0.07	0.00	0.00	1.34	-155.86	57.36	1242

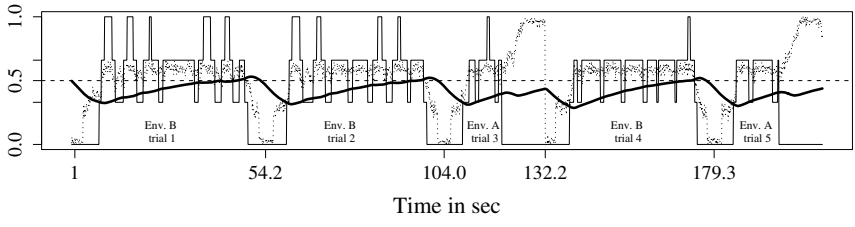


Fig. 2. The graph shows the output of the neuron that controls the sound (N_6, see continuous thick line), the floor sensor reading (F, see continuous thin line), the average values of the light sensor readings (A_1, A_2, A_3, A_4, see dotted line), during the first 5 trials of a 10 trials evaluation in which the robot did not make any error. The numbers on the x-axis, show at which point of the robot life-span a new trial begins.

(III) refers to the case in which the robot makes both errors I and II within the same trial. Similarly, in *Env B*, we looked at the performance of the robot in completing the task as mentioned above (*Succ.*), without committing any error. Four error types are possible: error (IV) refers to the lack of sound signalling, error (V) refers to the robot crossing the black edge of the band, error (VI) refers to the robot missing to reach the *dark* zone after having signalled; error (VII) refers to the case in which the robot makes error IV, V, and VI within the same trial. Furthermore, in *Env B* we also compute the offset (offset Δ) between the entrance position of the robot in the circular band and the position in which the robot starts to signal (see [9] for a description of how the offset Δ is computed). This measure accounts for the precision of signalling with respect to the time it

takes for the robot to complete a loop around the light. Offset Δ takes value 0° if the robot signals exactly after covering a complete loop of the circular band. Negative values of the offset Δ suggest that the robot signals before having performed a complete loop; positive values correspond to the situation in which the robot emits a tone after having performed a loop around the light.

The results of the post-evaluation, shown in Table 1, shed light on two aspects of our work: first, they give a quantitative estimate of the overall success rate of the evolved strategies; second, they provide elements to infer the behavioural strategies employed by our robots to solve the task. Concerning the percentage of success in both types of environment, the results are quite encouraging. Despite the complexity of the task, six runs out of ten—runs n. 2, 4, 5, 7, 8, 9—show a percentage of success (*Succ.*) in both types of environment higher than 80% (i.e., more than 2016 successful trials out of 2520). The strategies of run n. 2 and 9 are the most effective, with a percentage of success in both environments higher than 94%. The performance of run n. 4, 5, 7, 8 is mainly "blurred" by errors of type I, caused by a risk-taking behaviour, which led the robot to signal slightly before having completed a loop around a light—see the negative values of the average offset Δ. Among the less successful robots, the performance of run n. 10 is also disrupted by errors of type I. The bad result of run n. 6, and n. 2 is mainly due to crossing the black edge of the circular band (error type II). Run n. 1 is quite successful in *Env A*, but its performance is particularly bad in *Env B*. In view of its high error rate of type VII, we can conclude that this robot employs the strategy of never signalling in *Env B*, and of remaining on the circular band circuiting around the light. It is worth noticing that the two most successful runs (i.e., run n. 2 and 9) employ a risk-averse behaviour, since they have the tendency to signal slightly after having completed a loop around a light—see the positive values of the average offset Δ.

The graphs shown in Fig. 2 give us some hints on the mechanisms employed by robot run n. 9 (a) to control the sound, and (b) to switch from phototaxis to anti-phototaxis and vice-versa. As far as it concerns (a), Fig. 2 shows that the output of neuron N_6 increases if the robot is on the circular band. The output of N_6 crosses the 0.5 threshold—i.e., the sound is turned on—if the robot remains on the band for a time slightly higher than the time required to make a loop around the light. This can be inferred by the positive value of the offset Δ in Table 1. Concerning (b), the robot performs phototaxis as long as it does not perceive any sound. Fig. 2 shows that the perception of sound makes the robot change strategy—i.e., from phototaxis to anti-phototaxis. Once the robot is out of the band, the output of neuron N_6 starts decreasing. However, given the rate of change of the output of neuron N_6, the robot stops emitting a tone just after having reached the *dark* zone. Owing to this mechanism, the robot manages, by the end of a trial in *Env B*, to set the cell potential of neuron N_6 to a value which makes it possible for it (a) to approach the light at the beginning of the following trial, since no sound is emitted, and (b) to be in a "state" to be able to perform another discrimination.

4 Conclusions

In this paper, we have shown that a single dynamic neural network can be synthesised by evolution to allow an autonomous robot to successfully perform an iterated discrimination task, based on time-dependent structures. The results illustrated here are of particular interest because, to the best of our knowledge, this is the first study in which an autonomous robot manages to iteratively solve a complex non-reactive task without previous experience disrupting its decision making mechanisms. The performance of the best evolved robot was not disrupted by the sequence of presentation of the environments. However, the robustness of the evolved strategies with respect to other potentially more disruptive environmental changes, such as the dimensions of the circular band, and the *dark* zone, remains to be assessed.

Acknowledgements. This work was supported by the *ECAgents* project, funded by the Future and Emerging Technologies programme of the European Commission, under grant IST-1940. The authors thank Thomas Halva Labella and Vito Trianni for stimulating discussions and feedback during the preparation of this paper, and the three reviewers for their helpful comments. Marco Dorigo acknowledges support from the Belgian FNRS, of which he is a Research Director, and from the "ANTS" project, an "Action de Recherche Concertée" funded by the Scientific Research Directorate of the French Community of Belgium. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication.

References

1. C. Ampatzis, E. Tuci, V. Trianni, and M. Dorigo. Evolving communicating agents that integrate information over time: a real robot experiment. Technical Report TR/IRIDIA/2005-12, Université Libre de Bruxelles, 2005.
2. G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*. Cambridge University Press, Cambridge, UK, 2000.
3. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.
4. I. Harvey, E. A. Di Paolo, R. Wood, M. Quinn, and E. Tuci. Evolutionary robotics: A new scientific tool for studying cognition. *Artificial Life*, 11(1–2):79–98, 2005.
5. P. Husbands, T. Smith, N. Jakobi, and M. O'Shea. Better living through chemistry: Evolving GasNets for robot control. *Connection Science*, 10(3–4):185–210, 1998.
6. N. Jakobi. Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior*, 6:325–368, 1997.
7. E. A. Di Paolo. Evolving spike-timing dependent plasticity for single-trial learning in robots. *Philosophical Transactions of the Royal Society A*, 361:2299 –2319, 2003.
8. E. Tuci, M. Quinn, and I. Harvey. An evolutionary ecological approach to the study of learning behaviour using a robot-based model. *Adaptive Behavior*, 10(3-4):201–221, 2003.
9. E. Tuci, V. Trianni, and M. Dorigo. 'Feeling' the flow of time through sensory/motor coordination. *Connection Science*, 16:1–24, 2004.

Hysteresis and the Limits of Homeostasis: From Daisyworld to Phototaxis

James Dyke and Inman Harvey

Evolutionary and Adaptive Systems Group,
Centre for Computational Neuroscience and Robotics
Informatics, School of Science & Technology
University of Sussex, BN1 9QH, UK
j.g.dyke@sussex.ac.uk, inmanh@sussex.ac.uk

Abstract. All biological organisms must be able to regulate certain essential internal variables, e.g. core body temperature in mammals, in order to survive. Almost by definition, those that cannot are dead. Changes that result in a mammal being able to tolerate a wider range of core body temperatures make that organism more robust to external perturbations. In this paper we show that when internal variables are regulated via 'rein control' mechanisms, decreasing the range of tolerable values increases the area of observed hysteresis but does not decrease the limits of regulation. We present circumstances where increasing the range of tolerable values actually decreases robustness to external perturbation.

1 Introduction

In a biological context, the term homeostasis is applied to the inherent tendency in an organism toward maintenance of physiological stability. For example, mammals must maintain core body temperature to within a certain range if they are to survive. Mechanisms to maintain a minimum core body temperature could be shivering and reduction of blood circulation to the extremities. If core body temperature increases to the upper limits of this viability range, then sweating and dilation of capillaries will lower core body temperature. Following Ashby [1] we define such internal variables as essential variables. Furthermore we define the tolerance - the range of values that the essential variable must be maintained within - as the essential range. For example core body temperature in Homo sapiens must be maintained within the essential range of approximately 35-41 degrees Celsius.

In this paper we will argue that for systems that are regulated via 'rein control' (as discussed below), decreasing the essential range may not decrease the range over which homeostasis is performed. We will demonstrate that increasing the essential range may actually decrease robustness to external perturbation. Clynes [2] postulated that many physiological homeostatic processes operate on the basis of opposing control reins that each pull in a single direction in response to certain variables; in order to regulate for both upper and lower limits, two reins, two separate mechanisms

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 241 – 251, 2005.

are required. For a physiological application of rein control see Saunders et al [4] who employ Clynes' rein control analysis to understand the mechanisms responsible for the regulation of blood sugar levels in humans.

We will show that altering the essential range changes the area of hysteresis. Hysteresis will be observed in a system when the output or behaviour is bistable as an input parameter that represents some property of the system is changed through some range of values; when the input is increased through that range, the output is one function of the input, yet if the input decreases, the output is a different function of the input, thus tracing out a 'hysteresis loop'.

In order to explore these issues, analysis will be carried out on the behaviour of a simple two-box 'cable car' model that performs phototaxis. This model is based on the radically simplified Daisyworld model, as detailed by Harvey in [3]. The original Watson & Lovelock Daisyworld model [5] was intended to demonstrate the homeostatic properties of a planet that is covered with varying proportions of black and white daisies. Watson & Lovelock employ the Stefan-Boltzmann law to determine the temperatures of the daisies and bare earth. While such an approach involves a non-linear change in temperature with respect to absorbed energy, the relationship between the albedo and the temperature of a body is straightforward; given a fixed amount of short-wave radiation, the lower the albedo, the darker the body, the less radiation is reflected and so the higher the temperature. When the star that heats the planet is dim, the planet is cool. Black daisies, having a lower albedo than either white daisies or the bare earth absorb more of the radiated short-wave energy from the star and so will be warmer than either white daisies or bare earth. If the brightness or luminosity of the star steadily increases, then black daisies will begin to grow. As the proportional coverage of black daisies increases, the net albedo of the planet decreases. This raises the temperature and so increases the rate of daisy growth. The result of this positive feedback is a population explosion of black daisies and a sharp increase in the planetary and daisy temperatures. If luminosity continues to increase, the planet eventually becomes too warm for black daisies to be able to grow. Only white daisies are cool enough to survive as they reflect a greater proportion of the incoming solar radiation. In this way, the black and white daisies regulate the planetary temperature, keeping it within the essential range over a wider range of luminosities than would be the case with a bare lifeless planet.

Rather than formulating an abstract model of a homeostatic system, we instead follow the precedent of Daisyworld and present the cable car model in the form of a 'parable'. To that end, the simplification process begun in [3] is taken further. The cut-down 'toy' physics is reduced to simple linear responses to a light source position whilst the relationship between temperature and albedo is dispensed with. These further simplifications will aid investigation into the relationship between homeostasis, hysteresis, essential range values, and in particular demonstrate that increasing the essential range of the model decreases the area of observed hysteresis but does not increase the limits of self-regulation. Furthermore there are circumstances where increasing the range of tolerable values actually decreases the limits of self-regulation.

1.1 Organisation of Paper

The cable car model will be introduced and compared to Daisyworld in the following section. Both models are composed of two control reins, loosely coupled via their interaction with a shared external driving force. Results from the cable car model will be presented in Section 3. Section 4 will analyse the results. Section 5 concludes the paper.

2 The Cable Car Model

The model is based on the cable cars found in San Francisco. Unlike the systems used in the Alps and other mountainous regions, the San Francisco system consists of cables that are located under the road surface and connect to tram like cars. In our model a photovoltaic cell – a 'solar panel' – is located on the roof and supplies current to an electric motor which instead of being located in a winding house, is carried within the cable car itself. As the motor turns, it pulls in a cable that moves the car up the side of a valley. The output of the solar panel, and therefore the force that the motor produces, changes linearly with varying inclination from a moveable light source. When the light source is directly overhead, maximum output is produced and so maximum motor output is achieved. Deviations left or right by either the cable car or light source result in decreasing energy production. The range of light source locations that produce current in the solar panel we call the activation range and is analogous to the essential range of viable daisy temperatures in Daisyworld. It is assumed that the light source is so far away (e.g. the sun) that the energy input depends solely on relative angle to vertical, and any distance change is irrelevant. This does not make any substantive difference to the behaviour of the model but does allow easier analysis.

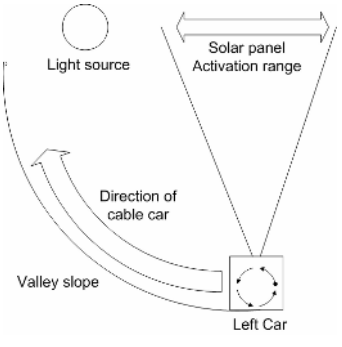


Fig. 1. As the light source enters the activation range of the solar panel, the motor rotates anti-clockwise, pulling on the cable which moves the car to the left and so up the valley slope. The gradient of the slope can be understood to increase non-linearly, e.g. the valley has a 'U' shape and so the car experiences an increasing 'resisting' force due to gravity as it moves further from its starting position.

As the light moves into the left-hand side of the activation range, the solar panel will begin to produce current. The motor will turn anti-clockwise pulling in the cable

and so move the car to the left. This will bring the light nearer the centre of the solar panel's activation range and further increase the motor output, and move the car further to the left, higher up the valley slope. The cable car has a dimensionless mass of 1 unit. As the car moves further from its starting position, the gradient of the slope increases and so the 'resisting' force due to gravity pulling the car back to the bottom of the valley, γ, increases:

$$\gamma = \eta \frac{X}{l} . \tag{1.1}$$

Where X is the position of the car in dimensionless x-units, l is the length of the slope and η measures the rate of increase of the resisting force as the car travels higher up the slope. For the simulations presented in this paper η was set to 1 and so γ increases linearly from 0 when the car is at the bottom, to 1 when the car is at the top of the valley.

α is the force produced by the motor. This was set to vary linearly from 0 to a maximum of 1 in response to the output of the solar panel:

$$\alpha = \mathrm{Max}\left[1 - 2\left(\left|X_{light} - X\right|\right)/\varphi,\ 0\right] \tag{1.2}$$

where X_{light} is the location of the light source and φ is the activation range of the solar panel. This produces a 'witches hat' shaped activation function that can be understood as a piece-wise linear version of the original Daisyworld parabolic growth function.

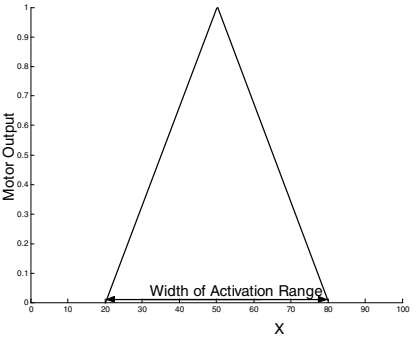


Fig. 2. The output of the solar panel is maximized and so motor output is greatest when the light is directly overhead. As the light moves away from this point, motor output decreases linearly.

The model is completed with the introduction of another cable car that moves up the opposite side of the valley.

The energy provided by the solar panel turns the motor clockwise and so the car moves to the right. A spring is attached between the cars. As the cars move apart, the spring is stretched and a force is exerted that pulls the cars back together. This force is found with:

$$F = \varsigma\left(X_{right} - X_{left}\right). \tag{1.3}$$

Where ς is the 'elasticity' of the spring and is parameterized from 0 (infinitely elastic, giving $F = 0$), to 1 (completely rigid so that both cars move as a single unit). It is important to note that F is based on the horizontal distance between the cars as measured in x-units. This will differ from the 'actual' distance due to the changing gradient of the valley slope. Such a difference does not make any substantive difference to the results, but does allow simpler computations. Table 1 lists the parameters of the cable car and Daisyworld models and allows a comparison of the two.

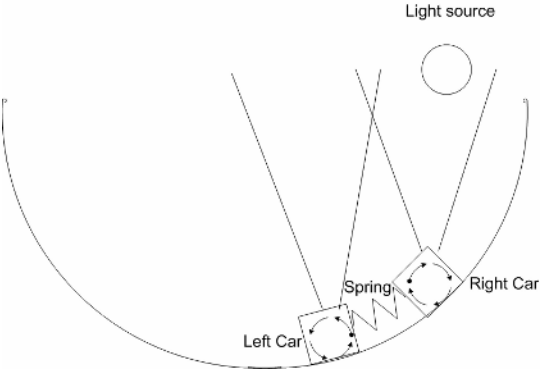


Fig. 3. The left car motor pulls to the left, whilst the right car motor pulls to the right. Depending on the strength of the spring, both cars will move independently or together. The solar panels remain pointing straight up irrespective of the orientation of the cable cars.

Table 1. A comparison of cable car model and Daisyworld parameters

Cable Car Model	**Daisyworld**
Motor output	Daisy coverage
Light location	Luminosity of star
Solar panel activation range	Range of viable growth temperatures
Left car	Black daisies
Right car	White daisies
Connecting Spring	Flow of heat from hot to cold

3 Results

Steady state values over a range of light source positions were found numerically by employing the following algorithm:

1. Calculate the energy produced by the solar panels from the angle of inclination of the light source and therefore the force of rotation of the motors.
2. Calculate the resistance pulling both cars back to the bottom of the valley.
3. Calculate the car's new positions as a sum of the motor output and resisting forces.

4. Calculate the force exerted by the coupling spring connecting both cars.
5. Move the cars towards a point midway between them in proportion to the coupling spring force.
6. Go back to 1.

For each time-step, the light source position remained fixed whilst steps 1-6 were iterated until changes in the position of the cable cars were vanishingly small as calculated at double floating point accuracy. In practice 10,000 iterations were sufficient. The width of the activation range was set to 20 x-units. The width of the valley was set to 100 x-units. The top of the left hand slope was located at x = 0, the bottom at x = 50 and the top of the right hand slope at x = 100. Motor output varied from 0 to a maximum of 1. η was set to 1 in order that γ varied linearly from 0, to a maximum resisting force of 1 when a car was at the top of either slope. The strength of the coupling spring was set to 5%, $\varsigma = 0.05$.

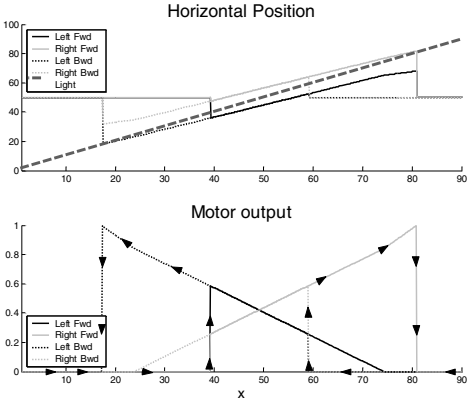


Fig. 4. Solid lines show the car's horizontal position (top plot) and motor output (bottom plot) when the light is moving left to right (forwards). Dashed lines show positions and motor output when the light is moving right to left (backwards). Arrows indicate the hysteresis loops.

As the light moves from left to right, both cars move immediately away from the bottom of the valley. As the light continues to move, the left car slowly moves back down the slope whilst the right car continues to move to the right until its motor output reaches its maximum. As the light moves further, it goes past the centre of the right car's activation range and so solar panel and motor output decreases. This moves the car to the left and so further away from the light source. The right car continues to move back down the slope until it is at rest at the bottom of the valley. As the light reverses direction the same situation occurs but with the right car steadily decreasing in motor output and the left car motor output steadily increasing to its maximum and then abruptly falling to zero. Such behaviour is similar to the growth curves and area of hysteresis observed in Daisyworld simulations in which there is initial rapid

growth, then steady decline of black daisies whilst the white daisies population slowly increases, peaks and then suddenly collapse.

3.1 Phototaxis as Homeostasis

The homeostatic properties of Daisyworld are assessed in terms of the system's ability to regulate global temperatures to within the range in which daisy growth in possible. This is achieved by maximizing the range of luminosity in which daisies are able to grow, and in particular the range in which both daisy types are present. The cable car model instead performs phototaxis. Phototaxis is defined here as the change in the cable car's position in response to changes in the light source's position, with the result that the position of the light source is maintained between the two cars. This is made more apparent if the location of the light source and cars are plotted over time with the light source taking a sinusoidal motion back and forth along the horizontal plane.

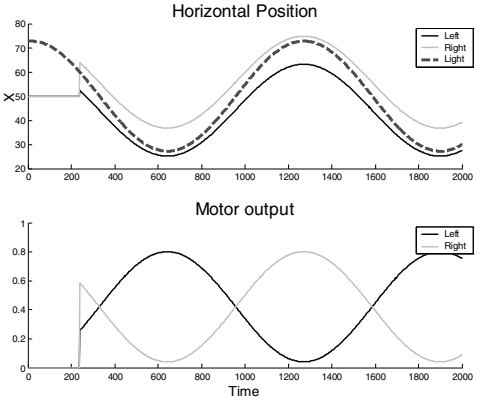


Fig. 5. Time is given in dimensionless units on the horizontal axis on both plots. The top plots shows both cars initially at rest at the bottom of the valley where x = 50. They are not activated by the light source until time step 210. The light source moves back and forth along the horizontal plane. The motor outputs and positions of the cars reflect this motion.

Hysteresis is observed when simulations are performed with the light source outside of the activation range of either cable car. Once the cable cars begin to move, they are able to track the light over the entire range of light source positions. If the light source begins within the activation range of either car, or if the right/left hand car is held at its maximum distance from the bottom of the valley, and the light is introduced from the right/left, then there is no period when the cars are inactive. This produces the perhaps counter-intuitive result that decreasing the activation range may not decrease the range of light positions over which phototaxis can be achieved. For example, the hat function could be transformed into a 'spike' function with the result that the light source must be directly overhead in order to produce solar panel output and so motor force.

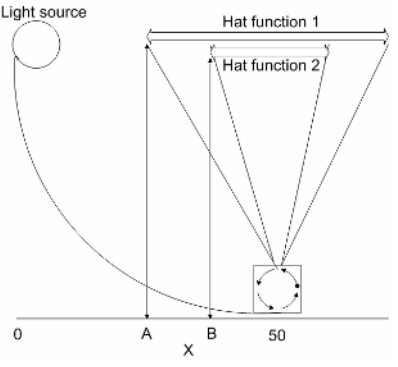


Fig. 6. Two solar panel hat functions for the left hand cable car are plotted. Hat Function 1 is wider than Hat Function 2. The cable car starts at x = 50 (the bottom of the valley). The light source starts at x = 0 (the top of the left hand slope) and moves to the right. Hat Function 1 will be activated at point A, when the light source enters the activation range of the solar panel, whereas Hat Function 2 will not activate until point B. Once the car is activated, if the light source were to reverse direction and move to the left, the limits of phototaxis would be the same for both hat functions.

For a single cable car, the limits of activation are not determined by the width of the hat functions rather their heights and the feedback that the cable cars exert. This feedback is a function of the strength of motor output and the sum of resisting forces pulling the car back towards the bottom of the valley. The limits of phototaxis $X_{\max}$, can be found with:

$$X_{\max} = \alpha_{\max} \frac{l}{\eta} . \tag{1.4}$$

where $\alpha_{\max}$ is the maximum possible motor output. If the left hand car starts at the bottom of a slope of length 50 and has a maximum motor output of 1 with η set to 1, the furthest left it can travel is 50 units. It is not necessary to specify the width of the solar panel activation range in order to determine $X_{\max}$.

3.2 Decreasing the Range of Phototaxis

In simulations with two cars, as the light moves back and forth, only one cable car is responsible for the light-following behaviour. This car is referred to as the 'active' car. The other car is pulled up the opposite slope by virtue of the connecting spring. This car is referred to as the 'passive' car. As the activation range increases, the motor output of the passive car increases and so the car pulls more strongly *away* from the light. This also increases the sum of resisting forces on the active car and so reduces the value of $X_{\max}$. Fig. 7. shows the effects of increasing the width of the activation function to 40 x-units and then decreasing it to 4 x-units.

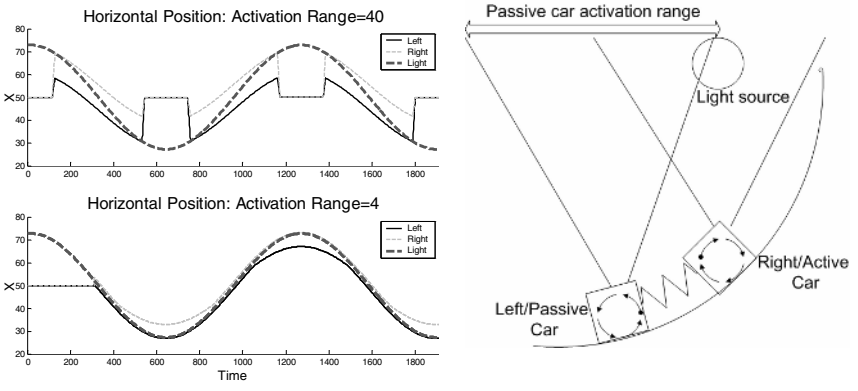


Fig. 7. In the left hand figure, the effects of increasing the activation range to 40 x-units are shown in the top plot, the effect of decreasing it to one tenth of this value are shown in the bottom plot. The range of phototaxis is significantly reduced when the activation range is increased to 40 x-units. When the activation range is reduced to 4 x-units, the distance between the cars and the light source decreases whilst the area of hysteresis increases. In the right hand figure, the effects of increasing the activation range to 40 x-units has resulted in the passive car solar panel being activated and so the left hand car pulling away from the light and so increasing the sum resisting forces acting on the active car. This decreases the limits of phototaxis.

4 Analysis

The strength of each feedback channel, how hard the motors pull, in part determines the range of external forcing over which the cable car model is able to perform phototaxis. What the channels are pulling against is just as important. Indeed it is the resistance that the cable cars experience as they travel up the valley slope that allows phototaxis to be performed and is also the cause of the observed hysteresis. To explain this will require a moment's anthropomorphising. The roots of hysteresis are found in the different cable car behaviours in response to the light source that is dependent on the direction from which it enters the activation range. For example, if the light source approaches the left car from the right hand side, the car attempts to 'run away' up the slope. It is light-phobic. This is referred to as behaviour A. If however the light source approaches from the left hand side, it is a light-phile. The car runs up the slope to 'meet' the light and then 'follows' it back down the slope. This is referred to as behaviour B. The response of the car is the same under both situations. As the cable car motor is only able to move the car up the slope, the moment the light is detected, the car attempts to 'escape' to the left. During behaviour B, this moves the light across the centre of the activation range, past the mid-point, coming to rest on the right hand side where the output of the motor is balanced by the resisting forces pulling the car back to the bottom of the valley. As the light continues to move to the right, the car follows as the motor output steadily decreases. Due to the coupling spring, the left hand car will continue to move to the right and travel up the opposite

slope as the right hand car performs its equivalent of behaviour A. Behaviour B is produced by the expenditure of the potential energy that the car stored when the light source first entered the activation range. It is the storing then release of this energy, the balance between γ and α as the cars move up and down the energy gradient that produces both behaviours.

As demonstrated in Fig. 7. increasing the essential range can have a dramatic impact on the limits of phototaxis. As the essential range of the cars increase, the sum of resisting forces on the active car increases as the passive car pulls more strongly away from the light source. The active car not only hauls the mass of the passive car up the valley slope, but has to drive against the passive car's motor which attempts to pull the cars in the opposite direction.

5 Conclusion

Homeostasis is a core concept for understanding real or artificial life. Rein control is a little known notion of key relevance to homeostasis. Here we have developed this to bring out the new insight that, under some circumstances, increasing the range of tolerable values for essential variables can actually reduce robustness to external perturbation. These are very general observations, but have been discussed in the specific case of the cable car parable.

The cable car model is an intentionally simple system, however the behaviour it exhibits is at times not straightforward. An analysis of the hysteresis observed within this system has focussed on the relationship between the essential range and regulation. It has been shown that decreasing the essential range increases the area of hysteresis whilst the limits of homeostasis remain unchanged. Results have been presented in which increases in the essential range, decreased the range of self-regulation.

In the cable car model, making changes that result in an increase in the area of hysteresis may be desirable if there is an energetic cost associated with moving cable cars. For example, the purpose of the cable cars could be to triangulate a randomly moving light source as part of a targeting system. If the system requires time to active and also to deactivate, then a series of activations followed by deactivations followed by activations would be inefficient. A much better strategy would be to wait until there is a good probability of the target remaining within range for enough time to fully activate the target system. This could be achieved by increasing the area of hysteresis via a decrease in the width of the activation function. The system would be more stable by reducing the range of values it is able to operate within. Once activated, the system would be able to track the target over the same range of values as a system that was configured with a wider activation range.

Hysteresis may be present in a wide variety of processes and mechanisms, for example, the differential operation of a servo moving in opposite directions, or the differential activation threshold of a neuron. In either situation, hysteresis can be regarded as an 'ecological affordance' in that the hysteresis is an implicit element of the agent's body or environment that may be harnessed to allow a desired or evolved set of outputs or behaviours. Analysis of the cable car model has provided insights

into the relationship between hysteresis and the limits of homeostasis. These insights may be applicable to not only cable car models and Daisyworld but many, if not all, natural or artificial homeostatic systems that operate on the basis of rein control.

References

1. W. Ross Ashby, *Design for a Brain: The Origin of Adaptive Behaviour*, Chapman & Hall Ltd and Science Paperbacks, 1965, 1972.
2. Clynes, *Cybernetic Implications of Rein Control in Perceptual and Conceptual Organization. NY Acad. Sci.* 156 (1969) pp629-670
3. Harvey I., *Homeostasis and Rein Control: From Daisyworld to Active Perception. Proceeding of the Ninth International Conference on the Simulation and Synthesis of Living Systems, ALIFE'9*, Pollack K,. Bedau M., Husbands P., Ikegami T. and Watson R.A. (eds) (2004) pp309-314. MIT Press, Cambridge MA
4. Saunders P.T., Koeslag J. and Wessels J.A., *Integral Rein Control in Physiology. J. Theo. Biol.* 194 (1994) pp 163-173
5. Watson A.J. & Lovelock J.E., *Biological homeostasis of the global environment: the parable of Daisyworld.* Tellus 35B (1983) pp284-289

Is an Embodied System Ever Purely Reactive?

Eduardo Izquierdo-Torres and Ezequiel Di Paolo

Centre for Computational Neuroscience and Robotics,
Deparment of Informatics, University of Sussex
{e.j.izquierdo, ezequiel}@sussex.ac.uk

Abstract. This paper explores the performance of a simple model agent using a reactive controller in situations where, from an external perspective, a solution that relies on internal states would seem to be required. In a visually-guided orientation task with sensory inversion and an object discrimination task a study of the instantaneous response properties and time-extended dynamics explain the non-reactive performance. The results question common intuitions about the capabilities of reactive controllers and highlight the significance of the agent's recent history of interactions with its environment in generating behaviour. This work reinforces the idea that embodied behaviour exhibits properties that cannot be deduced directly from those of the controller by itself.

1 Introduction

Is it possible to deduce the cognitive limitations of an embodied agent from the limitations of its internal dynamics? In particular, is an agent controlled by a reactive system able to perform only reactive behaviours? Questions like these compel us to look carefully at the meaning of now commonly used terms such as embodiment and situatedness, often discussed in the abstract, and try to unravel their implications for concrete systems.

A way of addressing the most specific of these questions is to build agents controlled by reactive systems and evaluate their performance in situations that require non-reactive responses. By a *reactive controller* we understand a system whose outputs are at each moment only determined by its current inputs. In order to make the problem non-trivial we need to define reactive behaviour in terms of the properties of the *task* and not the controller. For the purpose of this work we adopt a definition of reactive behaviour based on the classification introduced by Clark and Thornton [4] as the performance of a type-1 task, i.e., a task that requires the agent to exploit regularities which are directly apparent in the current input data. In robotics, obstacle avoidance is typically a type-1 task. In contrast, type-2 tasks require the exploitation of regularities which are 'hidden', or whose statistical visibility depends on some systematic recoding of the data. Accordingly, we will treat performance of a type-2 task as a form of *non-reactive behaviour*. Online learning is typically a type-2 task.

In this paper, evolutionary algorithms are used to design neurocontrollers for the behaviour of model agents which are then analysed dynamically. The goal is

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 252–261, 2005.

to explore the role of embeddedness in space and time in enabling non-reactive performance in systems that can only respond reactively. In particular, we investigate the relation between instantaneous response properties and time-extended performance in orientation tasks, and the time-dependence of responsiveness and 'decision making' in shape discrimination. In both cases, embodied agents exhibit properties that cannot be deduced directly from their reactive controllers. The dynamical analysis of these agents allows us to draw some general inferences about the danger of making *a priori* assumptions about the required properties of internal control mechanisms for a given task.

2 Background

We may find classical answers to our opening questions in criticisms of behaviourism. For instance, in Dewey's critique of the reflex-arc concept in psychology [5] it becomes clear that action is ongoing and stimuli can only have an effect on the behaving agent because the agent is capable of selecting them actively by the nature of its perceptual systems but also by the nature of its actions. The same point is compellingly made by Merleau-Ponty:

> "The organism cannot properly be compared to a keyboard on which the external stimuli would play and in which their proper form would be delineated for the simple reason that the organism contributes to the constitution of that form ... it is the organism itself – according to the proper nature of its receptors, the threshold of its nerve centers and the movement of the organs – which chooses the stimuli in the physical world to which it will be sensitive." (in [9] p.13)

We find similar views in Varela's work (e.g. [13]), where the emphasis is on cognition as embodied action wherein the world and the perceiver mutually specify each other. This is closely related to von Uexküll's functional circles [8], i.e. the formation of a closed unit between the 'perceptual' and 'effector' worlds that enables an agent to generate its own *Umwelt*. In robotics, Pfeifer subscribes to a related view [11], showing the importance of thinking in terms of sensorimotor coordinations. In recent years, we have seen concrete examples of these ideas at work in the area of autonomous robotics. For instance, Nolfi [10] provides examples in object size and shape classification tasks using reactive controllers and Scheier et al. [12] make similar points by studying object-constancy and focus-of-attention problems using hand-coded physical robots as well as evolved simulated agents. Implications of the embodied view in the context of biological neural networks have been summarised in [3].

What we propose to do here is to focus on the opportunities that the 'neural' architecture, body and environment offer to the system's controller. We will show how and why an embodied system can perform non-reactive behaviour (type-2 tasks) even when only endowed with a purely reactive controller. The interesting lessons will be in the details of how the agents work because they uncover the hidden assumptions about the capabilities of embodied and situated systems, even when their internal controllers are very simple.

3 Methods

We propose to study the role of the agent's situatedness using a set-up similar to the one presented in [1,2] with slight variations and extensions on the architecture and tasks. This model has been chosen for two reasons: its simplicity and its potential for sufficiently interesting behaviours that could be called minimally cognitive. As a modification to this set-up, the controller's architecture is made purely reactive. Two tasks are studied: an approach/avoid task which is made type-2 by the fact that the sensor array may be inverted, and a discrimination task to evaluate the reactivity of the agent at different stages. These are described in the following sections.

The agent has a circular body with a diameter of 30. The agent's 'eye' consists of six rays at $\pm\pi/12$, $\pm\pi/24$ and $\pm\pi/36$ from the centre. An intersection between a ray and an object causes an input to be injected into the corresponding sensory node, with the magnitude of the injected input inversely proportional to the distance to the object along that ray with a maximum length of 220. When the rays are at their maximum length, no input is injected, while the maximum input is injected for rays of zero length.

The agent can move horizontally as objects fall from above (Figure 1A) with horizontal velocity proportional to the sum of the opposing forces produced by two motors. The behaviour of the agent is controlled by a network of continuous-time recurrent neural nodes of the form:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=i}^{N} w_{ji}\sigma(g_j\,(y_j + \theta_j)) + I_i \qquad (1)$$

where y is the activation of each node, τ is its time constant, w_{ji} is the strength of the connection from the j^{th} to the i^{th} node, θ is a bias term, g is a gain, $\sigma(x) = 1/(1 + e^{-x})$ is the standard logistic activation function, I represents a constant external input (e.g. from a sensor) and N is the number of nodes in the network. In simulation, node activations are calculated forward through time by straightforward time-slicing using Euler integration with a time-step of 0.1.

The network architecture is bilaterally symmetric in the connection weights, biases and time-constants (unless otherwise specified). The architecture consists of six ray sensory nodes projecting to five inter-nodes, which in turn project to two motor nodes controlling horizontal motion (Figure 1B). All the sensory nodes share the same time-constant, bias parameter and gain parameter, while the rest of the nodes have a gain of 1.

The controller is made reactive by changing the connection weights between the inter-nodes to 0, fixing the time-constants for all nodes to 1, and modifying the time-step of integration to 1. In this way the state of any node in the network is fully determined by the pattern of inputs and cannot depend on previous internal states resulting in a discrete-time feed-forward artificial neural network.

The parameters for the neural controllers are evolved using a microbial genetic algorithm [7]. These are encoded as a vector of real numbers over the range [0, 1] (47 parameters for recurrent controllers and 26 for reactive ones). Offspring

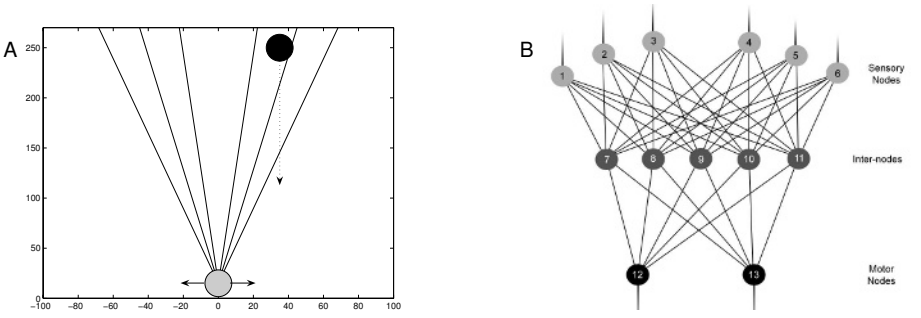


Fig. 1. Basic setup for the experiments. [A] The agent (gray circle) can move horizontally while objects (black circle) fall from above. The agent uses an array of 6 distal sensors (black lines). [B] The network architecture consists of six sensory nodes fully connected to five inter-nodes, which are in turn fully connected to two motor nodes.

of microbial tournaments are generated using recombination and vector mutation which consists of adding to the genotype a random displacement vector whose direction is uniformly distributed on the N-dimensional hypersphere and whose magnitude is a Gaussian random variable with 0 mean and variance 0.01. Population sizes of 100 and recombination rate of 0.9 are used. Genotypes are then mapped to network parameters using linear maps from $[0, 1]$ to $[-14, 14]$ for biases, $[-10, 10]$ for connection weights and $[1, 5]$ for the gain parameter while time-constants are exponentially mapped to $[e^0, e^4]$.

4 Orientation Experiments with Visual Inversion

In the first set of experiments, visually-guided agents are evolved to adjust their horizontal position so as to catch or avoid falling objects with normal and inverted vision. On inverting the visual field in the left/right direction an object that appears to the right of the agent will in fact be to its left. This task represents a type-2 problem, for it requires an agent to perform differently for the same stimuli depending on the context.

A simple evolutionary training regime is used. During an evolutionary evaluation 21 circular objects are dropped from the top of the environment straight down with an initial horizontal offset from the centre of the agent uniformly distributed in ± 50 and a fixed vertical velocity of -3. Following [6], this is repeated for objects of different diameter (ie. 26, 30 and 34). The whole process is then repeated using inverted vision, for a total of 126 trials in a fitness evaluation. At the start of each new trial node activations are initialised to zero.

The performance measure to be maximised is: $f = 1 - \sum_{i=1}^{N} d_i/N$, where N is the total number of trials and d_i is the horizontal distance between the centres of the object and the agent when their vertical separation goes to 0 on the ith trial (clipped to $max = 50$ and normalised).

Agents with a reactive controller that could orient to falling circles with normal and inverted vision turned out to be relatively easy to evolve. Over 20

evolutionary runs, the best evolved agent achieved a mean performance of 0.994 on the 126 evaluation trials after only 100 generations of evolution, with a mean performance of 0.992 using normal vision and 0.990 with inverted sensors on 10.000 randomly generated trials distributed with initial horizontal positions in $[-50, 50]$, and diameter and vertical velocity of the falling object between $[20, 40]$ and $[-4, -2]$, respectively.

Figure 2A shows the strategy used by the agent to catch falling circles with normal and inverted vision. Notice the opposed shading of the velocity fields in the two conditions. As the controller is reactive and symmetric, a stimulus produces instantaneous motor effects that are opposite in the case of normal and inverted vision, or put differently, a real and a virtual object in the same position produce exactly the same instantaneous effect. Yet the situated behaviour of the agent over time results in trajectories that catch both virtual and real objects.

In the normal condition, trajectories are attracted to the centre where the velocity field turns slightly divergent and then 'trapped' by the two bright regions of centring velocities which eventually converge on the object's horizontal position. In the inverted condition, central trajectories become convergent by the nature of the central field, and the rest of the trajectories initially move away from the centre only to be trapped in a different and wider convergent region, reaching the centre when the divergent fields no longer have the same effect. The evolved strategy involves taking advantage of the agent's multiple sensors and most successfully evolved agents relied on a very similar strategy.

Recurrent and time-based networks were evolved as well and analyses of the best evolved controller yielded the use of a similar strategy to that of the above analysed reactive network. Figure 2B shows the behaviour of the best evolved recurrent network over 20 evolutionary runs.

Agents with network architectures not constrained to be bilaterally symmetrical seemed to be relatively easier to evolve. The behaviour of the best evolved agent is shown in Figure 2C. The agent's strategy is to position itself sufficiently to one side of all falling objects, at which point real objects are seen with its right-sided sensors while virtual objects with its left set of sensors. The agent can then centre in on objects with opposite reactions according to which side they appear to be on. The result is a much simpler strategy for centring on both real and virtual objects.

A reactive agent needs to constantly engage with sensory stimuli in order to act which makes avoiding (as opposed to catching) falling objects with normal and inverted vision a counterintuitive task. Figure 2D shows the behaviour and dynamics of the best evolved reactive controller for such task. From the figure its strategy can be easily understood: under normal vision, the agent avoids objects that are far away and centres on objects that are relatively close. As a result, real objects get avoided as they start falling and disappear from the field of view early, while virtual objects are initially centred, reaching then the point were sufficiently closed objects get avoided.

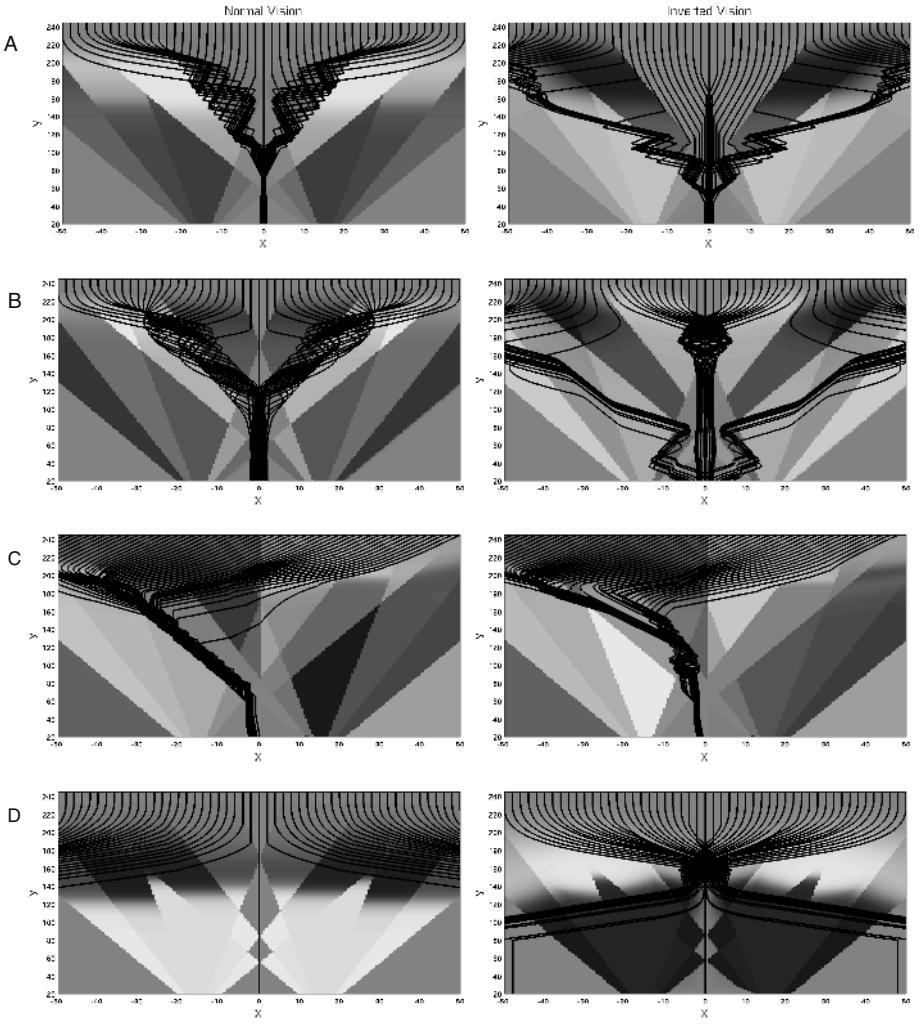


Fig. 2. Trajectories and steady-state horizontal velocity fields for normal (left) and inverted (right) vision for the best evolved agents for objects of diameter 30 and 51 different initial horizontal offsets for: [A] Circle centring task with a reactive symmetrical network; [B] Circle centring task with a recurrent symmetrical network; [C] Circle centring task with a reactive non-symmetric network; and [D] Circle avoidance task using a reactive symmetrical network. Trajectories are superimposed on differently shaded regions representing the long-term horizontal velocity that the agent would eventually adopt if an object were fixed at that location in its field of view at each point as a function of x the horizontal offset in relation to the agent and y the vertical distance of the object. Regions in which the agent is directed towards the object (centring regions) are bright, whereas those in which it is directed away (avoidance regions) are dark. The magnitude is represented by the intensity of the shade (mid-gray is no movement).

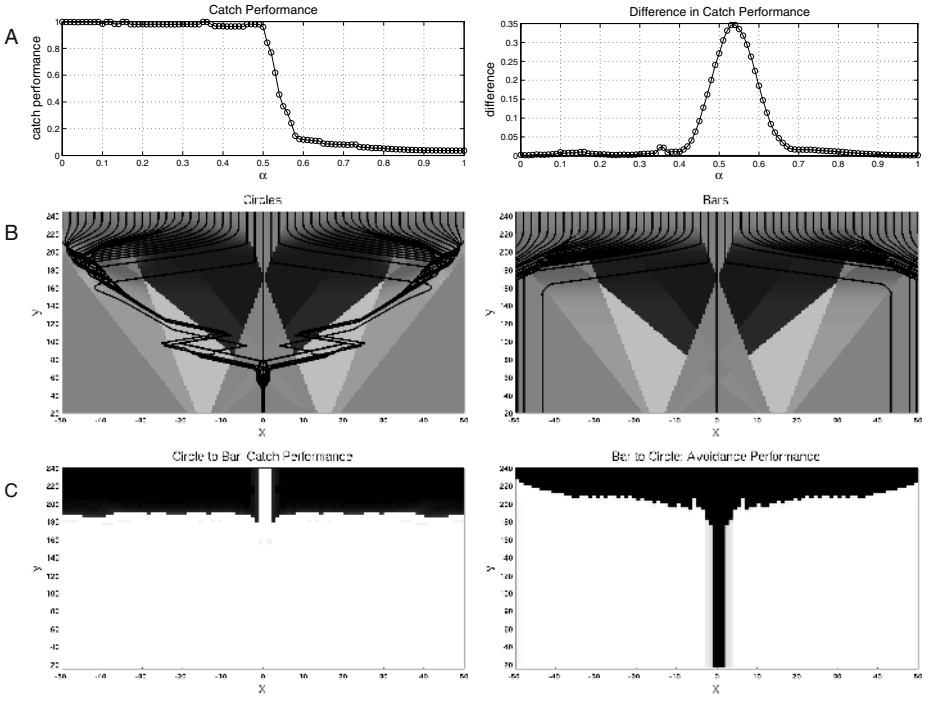


Fig. 3. [A] Demonstration of labelling and discrimination in the best evolved feedforward network. Average (left) and difference (right) in catch performance as a function of α. Each point represents the mean value for 101 trials at uniformly distributed horizontal offsets in ± 50. [B] Trajectories and steady-state horizontal velocity fields for the best evolved agent for circles (left) and bars (right). [C] Performance as a consequence from switching the object's identity from a circle into a bar (left) and viceversa (right) as a function of at different times on the final decision to catch or avoid in the best evolved agent. Each point represents the catch (left) and avoidance (right) performance as a function of initial horizontal offsets (x) and switching times (y). Bright means high performance. All figures are for objects of diameter 30 and vertical velocity -3.

5 Categorical Perception Experiments

In a second set of experiments, we explore agents that can discriminate between circular and horizontally oriented bars of different sizes using normal vision, catching the former and avoiding the latter in a similar task to the one explored in [2,6], in this case using a reactive controller. The evolutionary training regime used was similar to that used in the first set of experiments, with the only difference that half of the trials corresponded to circular falling objects and the other half to bar objects (as opposed to sensory inversion).

The performance measure to be maximised is: $f = p_i/N$, where N is the total number of trials and $p_i = 1 - d_i$ for a circular object and $p_i = d_i$ for bars, d_i is defined and normalised as above. Following [2], a class of parametrised hybrid

object that continuously interpolates between the bar and the circle is defined as $R(\alpha) = \alpha R_b + (1 - \alpha)R_c$, where $R(\alpha)$ is the radius of the hybrid object, and R_b and R_c are the radii (in polar coordinates) of the bar and circle respectively.

Over 20 evolutionary runs, the best evolved agent achieved a mean performance of 0.970 on the 126 evaluation trials after 100 generations of evolution, with a mean performance of 0.915 on 10.000 randomly generated trials from a broader set (initial horizontal positions between $[-50, 50]$, diameter size of the falling object between $[20, 40]$ and vertical velocity between $[-4, -2]$).

Two major defining characteristics of categorical perception are labelling and discrimination. In order to demonstrate these, the mean catch or avoid performance was plotted as the shape of the object changed between a circle and a bar (by modifying the parameter α). Figure 3A depicts the average catch performance as a function of α, a sharp sigmoidal curve with a transition from catching to avoidance behaviour at about $\alpha = 0.55$ is observed. Accordingly, the average difference in catch performance for α values that differ by 0.1 as a function of α shows a bell shaped function.

How are we to explain the behaviour of the agent? What sort of regularities does it exploit from the environment? The behaviour and steady-state dynamics of this agent are shown in Figure 3B. The evolved strategy involves positioning all falling objects at a particular angle of view where the difference between the two objects is maximised. This can be appreciated from dominating dark regions in the middle-top field of view of the steady-state velocity. At the point where the object is positioned close to the border of the agent's field of view, circular objects fall onto a very thin bright region of centring behaviour. This is further explained from a closer look at what the agent 'sees' (figure not shown), a circle never stimulates less than 2 sensors, while the bar stimulates only 1 sensor at one point, and this makes it move out of the sensor range.

An interesting question in the context of this paper is: at what point during the trial does the agent commit itself to catching or avoiding an object? What is expected from a reactive agent is a strong dependence, throughout the trial, between the shape of the object and the 'decision' to catch or avoid. This is explored by switching the shape of the object during the trial and observing the behaviour. In the absence of an internal state to 'retain' a previously made decision, one expects the decision to depend mainly on the shape after the switch.

Figure 3C shows the performance of the agent when catching a circle or avoiding a bar as a function of the horizontal offset and the distance where the switch from circle to bar, or vice versa, is introduced. The results are contrary to our expectations. Although the agent seems to be completely uncommitted during the initial movements, after passing a more or less well defined 'decision line' it becomes largely committed to either catching or avoiding even if the shape is changed afterwards. The 'decision process' is very much a discrete event that occurs in the early stages of the trial.

The intuition goes wrong because it generalizes from the instantaneous effect of a pattern of stimuli on a reactive controller to the time-extended and situated behaviour of the agent. If, as explained above, discrimination is achieved by a

particular correlation between object shape and angle of sensing chosen by the agent, and if after that event, independently of the decision made, the agent is already positioned in either a neutral or a centring velocity field, then any subsequent change of shape will be ignored. This is because behaviour does not depend on the objective shape of the stimulus but more precisely on the sensorimotor correlation between object and agent.

6 Discussion and Conclusions

This paper has demonstrated the evolution of embodied agents with reactive controllers for visually guided-orientation with sensory inversion and object discrimination. Although the tasks are interesting in themselves, the point of this paper is not to generate novel behaviour but to probe the intuitions concerning the capabilities of reactive controllers.

This work provides concrete examples showing how an embedded system is never purely reactive. From the example of shape discrimination, we show that the evolved agent will exploit state arising from its interaction with the environment and exhibit commitment to a decision. Agents modify their position with respect to other objects in the environment and, thus, partially determine the sensory patterns they will receive in the next time-step, thereby providing a concrete example of an agent creating the form of the stimulus by its manner of offering itself to actions from the outside, paraphrasing Merleau-Ponty.

For the visual inversion experiment the agent relies on following time-extended dynamics. As the state of the controller depends only on the pattern of inputs, the velocity fields for the normal and inverted conditions are point-by-point, opposed to each other. Which does not mean that the final state of the whole trajectory will be different in each case. This prompts an important conclusion: *the limitations of reactive controllers (or generally any given class of controllers) during extended periods of time in a situated system cannot be trivially deduced from the instantaneous, snapshot limitations of the same controllers.* Inversion of the sensory array produces an instantaneous reversal of velocities, and yet it results in a conservation, not a reversal, of the end-state of behaviour.

We illustrate some of the implications of reducing the assumptions about the necessary design of the agent's internal control mechanisms. In the visual inversion scenario, losing the symmetrical constraints allows the agent to *redefine* the problem into an easier one: catching objects that fall only to one side of it.

We do not deny the importance of an agent's internal dynamics in the generation of behaviour. It may, nevertheless, be the case that agents with internal dynamics exploit first the strategies available from its situatedness alone. In the visual inversion experiments agents with internal dynamics have the potential to solve the task in a variety of ways – for example, learning online which way around the sensors are wired up – and then acting accordingly. It is likely, however, that the evolved agents make use of the simpler embodied strategies first, as is shown from the evolved recurrent time-based network.

In summary, *a reactive controller in an embodied system doesn't imply reactive behaviour*: there is a difference between the local, instantaneous state defi-

nition of reactivity, and the behavioural definition, i.e., not being able to solve type-2 problems such as approach or avoidance under normal and inverted vision. As a result, whether an embodied agent will behave reactively (i.e., whether it will only be capable of performing behaviours of type-1) cannot be fully determined by the presence of a reactive controller.

The strategy proposed by minimally cognitive tasks for a critical examination of internal representation is straightforward: evolve agents on tasks that are 'representationally-interesting', then examine whether the agent is using representations [1]. In this case, no internal state is available for manipulation, thus, trivially, nothing is 'internally represented', yet behaviours such as commitment to discrete decisions on a categorisation task can still be demonstrated.

Future work will explore extensions to the capabilities of reactive controllers in a variety of directions. In general, it will be interesting to continue to relax *a priori* assumptions and consider how dynamical, bodily, and environmental constraints can transform 'cognitively hard' problems into easier ones. Some of these directions include: the effects of physical inertia, non-symmetrical architectures and noisy inter-node interactions.

References

1. Beer, R.D.: Toward the evolution of dynamical neural networks for minimally cognitive behavior. In Proc. of Simulation of Adaptive Behavior. MIT Press (1996).
2. Beer, R.D.: The dynamics of active categorical perception in an evolved model agent. Adaptive Behavior **11(4)** (2003) 209–243.
3. Chiel, H.J., Beer, R.D.: The brain has a body: Adaptive behavior emerges from interactions of nervous system, body and environment. *Trends in Neurosciences* **20** (1997) 553–557.
4. Clark, A., Thornton, C. Trading spaces: computation, representation and the limits of uninformed learning. Behavioral and Brain Sciences **20** (1997) 57–90.
5. Dewey, J.: The reflex arc concept in physchology. Psychological Review **3** (1896) 357–370.
6. Di Paolo, E. A., Harvey, I.: Decisions and noise: the scope of evolutionary synthesis and dynamical analysis. Adaptive Behavior, **11(4)** (2003) 284–288.
7. Harvey, I.: The microbial genetic algorithm. Unpublished manuscript (1996).
8. Lashley, K., Schiller, C.: Instinctive behavior: the development of a modern concept. International University Press (1957).
9. Merleau-Ponty, M.: The Structure of Behavior. Beacon Press (1967).
10. Nolfi, S.: Power and the Limits of Reactive Agents. Neurocomputing **49** (2002) 119–145.
11. Pfeifer, R., Scheier, C.: Understanding Intellingence. Cambridge, MA. MIT Press (1999).
12. Scheier, C., Pfeifer, R., Kunyioshi, Y.: Embedded neural networks: exploiting constraints. Neural Networks **11** (1998) 1551–1569.
13. Varela, F., Thompson, E., Rosch, E.: The Embodied Mind: Cognitive Science and Human Experience. Cambridge, MA. MIT Press (1991).

t for Two
Linear Synergy Advances the Evolution of Directional Pointing Behaviour

Marieke Rohde and Ezequiel Di Paolo

CCNR — Centre for Computational Neuroscience and Robotics,
University of Sussex
{m.rohde, ezequiel}@sussex.ac.uk

Abstract. Motor synergies, i.e. systematic relations between effectors, have been first proposed as a principle in motor control by N. Bernstein in 1935. Thereafter, his idea has inspired many models of motor control in humans and animals. Recently, "linear synergy", i.e. a linear relation between the torques applied to different joints, was reported to occur in human subjects during directional pointing movements [4]. In this paper, results from experiments in evolutionary robotics are presented to explore the concept of synergies in general and the role of linear synergy in the organisation of movement in particular. A 3D simulated robotic arm is evolved to reach to different target spots on a plane. Linear synergy is not found to be an outcome of the evolutionary search process, but imposing linear synergy as a constraint on artificial evolution dramatically improves evolvability and performance of evolved controllers.

1 Motor Synergies

The centipede was happy quite,
Until the toad in fun
Said "Pray, which leg goes after which?"
Which worked his mind to such a pitch,
He lay distracted in a ditch,
Considering how to run.

This anonymous ditty expresses nicely what the Russian physiologist N. Bernstein, a pioneer in biomechanics and anticipator of ideas of cybernetics, has identified as the degrees of freedom (DoF) problem as early as 1935 (English translation 1967 [3]): if we conceive of the central nervous system (CNS) as a homuncular control organ that determines the state of all actuators at any point in time, the control problem it has to solve is of inconceivably high dimensionality. Describing human or animal motion in terms of joint kinematics already involves a large number of DoFs (e.g. 7 in moving an arm), but if motor control is thought of in terms of individual muscles, or even motor neurons, the number of DoFs to be controlled just for moving an arm quickly exceeds four digits [12].

Not only is the problem space intractably large, but it is also *redundant* with respect to the outcome of an action, a condition that Hebb has termed

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 262–271, 2005.

"Motor Equivalence" ([6], p. 153ff). Yet another difficulty in practising motor control is *context conditioned variability* ([12], p. 246ff): the effect of a motor command is sensitive to the anatomical, mechanical and physiological context of the interaction of an agent with its environment, e.g. limb positions, passive dynamics or the state of the peripheral nervous system.

Motor synergies, i.e. *systematic relations between effectors*, are Bernstein's solution to the DoF problem. Just as the driver of a car, due to the linkage of the two front wheels, can determine both their positions by using just one steering wheel, mutual constraints in an organism's motor system could serve to build functional subunits, thereby reducing the effective number of DoFs in a motor task. Although this idea of motor synergies has greatly influenced research in motor behaviour (e.g. [1,5,8,10]), it is not free from practical and conceptual problems: is explaining the CNS as the driver of a bodily car much easier than explaining the whole system? Where do synergies come from, what is a good synergy, what mechanism controls their development and maintenance and the interaction between synergies acting in parallel? Does the evidence that "the context in which a motor task is executed strongly influences its organization" ([14], p. 74) not contradict the idea of low level synergies organising movement primitives in a constrained and automated manner?

Motor synergies are evident in human and animal behaviour (for a summary of findings see e.g. [13]). In situated and embodied approaches, which typically reject homuncular explanations, it is not clear what functional role synergies could bear, as without a homunculus, there is no need for "low level" mechanisms to interprete abstract orders in terms of bodily coordinates. It is exactly the puzzle of their purpose that make synergies an interesting phenomenon to explore from a situated and embodied perspective, in particular within an evolutionary robotics framework. The following are some of the questions that can be addressed: Where do synergies come from? Under which circumstances do they arise? Are synergies epiphenomenal to a structured agent environment interaction or implemented in the control architecture of an evolved agents? In the reverse direction, thinking in terms of synergies can also enrich our understanding and practice of evolving artificial agents: in the endeavour of staying tractable, many evolutionary robotics experiments seem to *implicitly presuppose the existence of synergies*: control architectures are frequently not redundant in DoFs and the role of actuators is rarely subject to context conditioned variability, as in biological organisms. Any comparison of the processes realising a certain behaviour in a robot and those realising the same behaviour in a human or animal will always rely on synergy–like low level processes to compensate for such idealisations. To study the impact of idealising assumptions can improve our use of evolutionary robotics as a method for the study of intelligent behaviour.

In this paper, the effects of motor synergies and redundancy in DoFs in an exemplary motor control task are investigated by performing a systematic comparison of evolvability and evolved behaviour in different kinds of controllers. The directional pointing task chosen is inspired by empirical experiments where

linear synergies were observed in human subjects. If linear synergies are beneficial to the organisation of the modelled task, their existence should lead to an improvement in either performance or evolvability. The results are analysed with respect to methodological issues in evolutionary robotics and put into relation with the empirical findings by which they are inspired.

2 Experiments in Directional Pointing

The experiments presented in this paper are inspired by findings reported in Gottlieb et Al. [4] on the linear relation between the torque applied to the shoulder and the torque applied to the elbow during human directional pointing in the sagittal plane. This principle of linear synergy could not be expected from the nature of the task and does not appear to be the outcome of a learning process, since infants in the pre–reaching period already apply it, even though their attempts to grasp an object are unsuccessful [15]. The role linear synergy plays in the realisation of pointing movements remains mysterious.

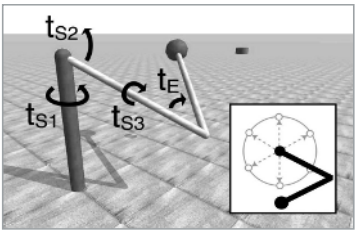


Fig. 1. The simulated robotic arm. Inlay: Plan view of the task.

Following these empirical findings, a simulated robotic arm is evolved to reach to six different targets in the horizontal plane. The arm is simulated in the C++ open source physics simulation library Open Dynamics Engine (ODE, [11]). It consists of a forearm, an upper arm (each two units long) and a spherical hand (Fig. 1). The six target points are spread evenly with uniformly distributed noise $\epsilon\,[0, \frac{1}{6}\pi]$ on the circumference of a circle with a radius of 1.25 around the unitary starting position of the hand at an elbow angle of 60° (Fig. 1, inlay).

Every DoF is controlled by applying a torque t_i to a joint j_i. In order to test the effect that the number of DoFs has on the task, experiments are run on a planar (**2D**) condition (one DoF in each, elbow (t_E) and shoulder (t_{S1})) and a three dimensional (**3D**) condition (one DoF in the elbow (t_E) and three DoFs in the shoulder: rotation in the horizontal plane (t_{S1}), lifting/lowering (t_{S2}) and rotation about the arm length (t_{S3}), Fig. 1, left). Joint stops are applied following the human example. In order to keep the task complexity comparable in the **2D** and the **3D** condition, deviation of the hand from the horizontal plane in the **3D** condition is made impossible. Thereby, the model loses biological plausibility, but not the suitability to explore the principrole of redundant DoFs in a motor task in principle. Dry friction is applied at all joints, gravity is not modelled.

The control networks evolved are continuous time recurrent neural networks (CTRNNs, e.g.[2]) with an input layer projecting to a fully connected hidden layer, which again projects in a feed forward fashion to the output neurons. The dynamics of neurons n_i in a CTRNN of N neurons are governed by

$$\tau_i \frac{da_i(t)}{dt} = -a_i(t) + \sum_{j=1}^{N} w_{ij}\sigma(a_j(t) + b_j) + I_i \quad (1)$$

where $\sigma(x) = \frac{1}{1+e^{\ x}}$ is the logistic function, I_i is the external input and a_i is the activation of n_i. The weights $w_{ij} \epsilon [-7, 7]$ from n_j to n_i, the bias $b_i \epsilon [-3, 3]$ and the time constant $\tau_i \epsilon [0.01, 1.77]$ are set by a genetic algorithm (GA).

Three different conditions are investigated for both the **2D** and **3D** set-ups (network architectures: see Fig.2, number of evolved parameters: see Table 1).

Table 1. Number of parameters evolved

	UC	**SB**	**FSa**	**FSb**
2D	109	75	53	46
3D	161	115	62	83

In the unconstrained (**UC**) condition, torques t_i are given by $t_i = MG_i \cdot (a_x - a_y)$ where $MG_i \epsilon [0.1, 30]$ is the evolved motor gain and $a_{x,y}$ is the activity of the two antagonistic motor neurons dedicated to the control of the joint j_i. The network has sensory neurons for the angular position of each DoF, one sensory neuron for the pointing direction $\theta \epsilon [0, 2 \cdot \pi]$ and six hidden neurons.

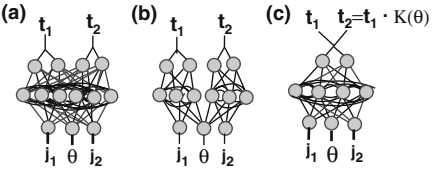


Fig. 2. Network architectures for the **2D UC** (a), **SB** (b) and **FS** (c) condition

In the "split brain" (**SB**) condition, controllers for each joint are co–evolved. They share the directional input neuron, but are only fed the corresponding joint angles j_i and have just three hidden neurons each. Comparing results from the **SB** and the **UC** condition is interesting with respect to role of a *neural basis for synergies*: The lack of connections between the controllers for each joint does not rule out the formation of synergies. Regularities in activation could in principle be mediated through the environment. Discovering such synergies would pose a clear challenge to homuncular explanations of synergies.

Finally, in the forced synergy (**FS**) condition, only the torque t_E to the elbow is generated by a CTRNN, the other joint torques t_{Sj} are scaled as a linear function $t_{Sj} = K_j \cdot t_E$ where K_j is constant within a pointing movement, but varies systematically across trials with the desired pointing direction: $K_j = f(\theta)$. Two different functional representations are used, in a condition called **FSa**, $K(\theta)$ is a simple linear function

$$K_j^a(\theta) = k_j^1 \cdot \theta + k_j^2 \quad (2)$$

with $k_j^i \epsilon [-4, 4]$ set genetically. In the condition **FSb**, $K_j(\theta)$ is represented by a radial basis function (RBF) network with Gaussian RBFs

$$K_j^b(\theta) = \sum_{i=1}^{4} w_{Ri} \cdot e^{-\frac{\delta^2}{2\ \Delta^2}} \quad (3)$$

where $\delta = c_i - \theta, d\,\epsilon\,[-\pi, \pi]$ is the difference in direction between the evolved RBF center $c_i\,\epsilon\,[-\pi, \pi]$ and the target direction θ. The width of the Gaussian RBF $\Delta\,\epsilon\,[0.5, 1.5]$ and the weights $w_{Ri}\,\epsilon\,[-4, 4]$ are set genetically.

All sensory inputs in all conditions are multiplied by a genetically set sensor gain $SG_i\,\epsilon\,[0.1, 20]$ The behaviour of the network is simulated using Euler integration with a time–step of 0.01 time units, the same time step as used in the simulation of the arm in ODE. Trials are run for $T\,\epsilon\,[2000, 3000]$ time steps.

The parameters of the control networks are evolved in a population of 30 individuals with a generational genetic algorithm with real–valued genes $\epsilon\,[0, 1]$, truncation selection ($\frac{1}{3}$), vector mutation ([2]) of magnitude $r = 0.6$ and reflection at the gene boundaries. All values are mapped linearly to the target range, apart from the sensor gains SG_i, the motor gains MG_i, the time constants τ_i, the absolute values of the coefficients $|k_i|$ and the absolute values of the RBFN weights $|w_{Ri}|$, which are mapped exponentially.

The fitness $F(i)$ of an individual i on a target spot j is given by

$$F_j(i) = 1 - \frac{d_j(T, i)^2}{d_j(0, i)^2} \tag{4}$$

where $d_j(t, i)$ is the distance of the hand from the target spot j at time t for individual i. Networks for all conditions are evolved with either incremental evolution, where the next clockwise target spot is added to the evaluation once the average performance of the population exceeds $\bar{F} = 0.4$ (starting with two positions) or on all six target spots right from the start. The evaluation of a network i on n target spots is given by

$$F(i) = \sum_{j=1}^{n} F_j(i) \cdot 2^{-(j-1)} \cdot \frac{1}{\sum_{j=1}^{n} 2^{-(j-1)}} \tag{5}$$

where $F_j(i)$ gives the fitness on the j^{th} worst evaluation trial for individual i, which gives more weight to worse evaluations and thereby rewards the generalisation capacity of the evolved networks.

3 Results

3.1 Number of Degrees of Freedom

Even though the investigated pointing task is already redundant in the **2D** condition because of the infinite possible trajectories leading the hand to the target position, the liberation of movement by providing two additional DoFs in the **3D** condition results in an enormous increase in performance in all network architectures: the number of target spots reached is much higher (Fig. 4).

Although the temporal organisation of movement can vary in **2D** networks, they do not have other possibilities for solving the task than to bring the two planar joints in the appropriate end positions. The **3D** networks, contrariwise, make excessive use of the additional DoFs and exploit passive dynamics (in this

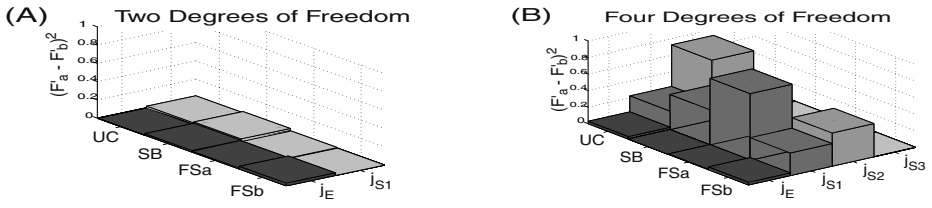


Fig. 3. Squared difference in normalised performance as individual joints are free to move but not driven (F'_a) or blocked (F'_b) in exemplary **2D** (A) and **3D** (B) networks

case joint motion due to environmental forces) to exhibit a multitude of strategies for solving the pointing task. An invariant in the behaviour of the **UC** and the **SB** networks was that their solutions involved turning the arm along its length to one of the joint stops — apparently, the positions thereby reached are more suitable for evolutionary search than the original starting position.

The performance of evolved controllers i was tested by selectively "anaesthetising" DoFs ($F'_a(i)$), i.e. not applying the motor torques, but allowing passive joint motion, or blocking them ($F'_b(i)$), i.e. not allowing joint motion at all. Figure 3 shows the squared difference in performance $(F'_a(i) - F'_b(i))^2$ between those conditions: where enabling passive dynamics to work on the anaesthetised DoFs does hardly make a difference in the **2D** condition (Fig. 3 (A)), it has a noticeable impact on performance of all **3D** networks. This better behavioural stability of the *3D* networks seems to be consequent to exploitation of the closed sensorimotor loop by mediating forces through the environment.

These findings illustrate how suppressing passive dynamics and endowing an agent with the minimally required sensorimotor system for a task can bias and limit evolved behaviours and even hamper evolution, even though dimensionality of the search space is reduced.

3.2 Forcing Linear Synergy

Probably the most significant finding from the presented experiments is the dramatic advantage that the **FSa** and **FSb** networks have in performance and evolvability over both the **UC** and **SB** networks [1]. Figure 4 shows how, in incremental evolution, the **FSb** networks advance to the next goal twice as many times as the non–**FS** networks. With twice as many generations, the non–**FS** networks come closer to, but never reach the level of performance of the **FS** networks. The **3D FSb** network is the only one that succeeds in solving the entire problem space; average performance of best individuals after 1000 generations is 0.65. Non–incremental evolution led to qualitatively similar results, i.e. quicker evolution of networks with much higher performance under the **FS** conditions.

Where it could be argued that the RBFN is simply a very suitable representation for a scaling function in this task, this is certainly not the case for

[1] Evolvability is simply conceived of as the level of performance to which a controller evolves reliably in a given number of generations using the described GA.

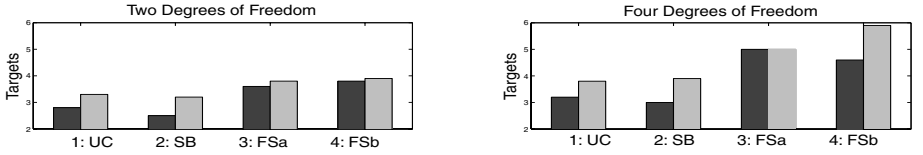


Fig. 4. Average number of starting positions reached in incremental evolution after 100 (dark) and 500 (light) generations across ten evolutionary runs

a simple linear function, particularly in the **2D** case, where the sensorimotor system of the robot is already so restricted that adding this additional harsh constraint makes it impossible to generate a controller that masters the task, not just because of the singularity at $\theta = 2\pi$. To rule out the possibility that the **UC** and **SB** controllers simply could not cope with the presentation of the input direction as a scalar neural input, a more "CTRNN friendly" set–up was tested, where controllers were provided with six different input neurons for the different target spots and no noise applied to θ, but neither in the **3D** or the **2D** condition could they go beyond three targets within 1000 generations.

Evolving controllers for directional pointing under the constraint of linear synergy could be shown to significantly improve both evolvability and performance of the resulting networks. Even when trying to remove biases in the experimental set–up that could give the **FS** networks a task–specific advantage, this advantage persisted. The division of control into scaling function and generation of motor signal is in some way suitable for evolutionary search. The nature of this benefit, however, is unclear, as analysis of the ruggedness of the fitness landscape around successfully evolved individuals did not provide an explanation[2].

3.3 Evolved Synergies

As for the linear synergies $K_i(\theta)$ evolved, no general pattern could be observed. Figure 5 (A) depicts an exemplary evolved RBFN in the **3D FSb** condition: the systematicities with which the scaling constant $K_i(\theta)$ varies in the different DoFs do not stand in an obvious relation. The displacement and overlap of peaks in these functions explain the diversity of behavioural strategies for different domains of θ observed in the **FSb** networks: For different targets, different DoFs are predominant in the realisation of the task.

If the principle of linear synergy was a characteristic of a good solution to the pointing task, we would expect an increase of linearity in torque relation as the performance of the **UC** and **SB** networks increases. Figure 5 (B) shows the sum of squared error from linear synergy in the **2D** and **3D UC** and **SB** controllers in the best individuals across five evolutionary searches: Where a slight

[2] Successful individuals were mutated in random direction with increasing magnitude of mutation r to compare their decay in performance, which can indicate the slope and ruggedness of the local fitness environment. On average, there were no discrepancies between the different conditions in the test, and large variance of decay profile between controllers within the same condition at a comparable level of performance.

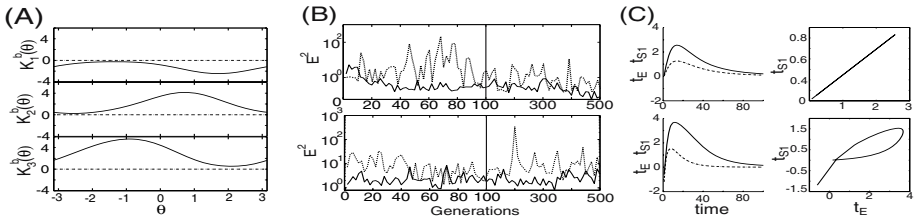


Fig. 5. (A): An exemplary evolved RBFN for a **3D FSb** network. (B): Sum of squared error from linear synergy across generations in the **2D** (top) and **3D** (bottom) networks, solid: **UC**, dashed: **SB**, average across five evolutions. (Note non–linear scales) (C): A **2D UC** network (bottom) applying a similar strategy as a **2D FSb** network (top).

tendency to get closer to linear synergy as performance increases is noticeable in the **2D** networks, in the **3D** networks, linear synergy and performance appear to be completely unrelated. It is remarkable that, on average, the **SB** networks act much less in linear synergy (note logarithmic scale), even though variance in the **SB** networks is much higher. Linear synergies without a neural basis were not evolved. From the mere architecture of the **SB** networks, this tendency to not simultaneously activate joints can be expected. However, a remarkable discrepancy between behavioural breakdown when anaesthetising ($F'_a(i)$) or blocking ($F'_b(i)$) degrees of freedom (see Sec. 3.1) was observed in the **3D** condition: this discrepancy suggests an involvement of the environment in the evolved solution, the prerequisite for establishing synergy without a neural basis. Being more disposed to linear synergy does not give the **UC** controllers an advantage, suggesting that the magnitude of deviation from linear synergy is not essential.

Figure 5 (C) shows, how a **2D UC** network applies a very similar strategy for solving a task, but slightly deviates from linear synergy, which leads to a loop in the t_E/t_{S1} map. On a performance level, this does not mean a disadvantage.

A direct correlation between the level of performance and the level of linear synergy is not evident, particularly in the **3D** networks, where the deviation from linear synergy in the random initial populations is maintained throughout evolution. The principle of linear synergy is not a priori a good strategy for solving the pointing task — the advantage linear synergy offers for the evolution of directional pointing behaviour is through constraining the search space.

4 Discussion

A series of evolutionary robotics experiments has been conducted to shed light on the role of linear synergy in a directional pointing task. Linear synergies could not be found to be the outcome of an unconstrained evolutionary search process. Furthermore, disconnected controllers for the different joints did not have a significant evolutionary disadvantage compared to monolithic networks controlling both joints, suggesting that the mere possibility of implementing constraints between effectors in a network does not provide a selective advantage. However, imposing the constraint of linear synergy boosts the search process,

even with impoverished linear scaling functions but, more significantly, if an RBFN represents the systematic variation of the scaling constant. The benefits of passive dynamics and redundant DoFs became clear during the analysis.

For the investigated task, both a complication (i.e. adding more DoFs) and a restriction (i.e. forcing linear synergy) of the search space have provided independent evolutionary advantages. Thus, improving evolvability is not a matter of scaling up or scaling down the search space, but of *reshaping the fitness landscape*. As tasks and robotic platforms become more complex, evolutionary robotics must produce appropriate reshaping techniques to scaffold the search process and thereby solve the "bootstrap problem" ([9], p. 13). Such a reshaping always means biasing evolutionary search. A lesson that may be learned from the present results is that biologically-inspired biases can not only help to make a stronger connection between models and empirical findings, but they can also be beneficial from an engineering perspective.

A generalisation about the evolvability of the applied technique, however, is not justified. Apart from technical difficulties, such as defining where one movement starts and another one ends or formalising the context of the task as a set of variables according to which a scaling constant K_i can vary, in many tasks a linear relation between effectors will be disadvantageous: for instance, a two-wheeled robot doing obstacle avoidance will obviously rely on an ongoing change in the relation between the effectors. A hypothesis put forward in this paper is that the principle of linear synergy will provide an evolutionary advantage in tasks that are not primarily reactive or if the motor system of a robot is redundant in DoFs. Otherwise, the imposed restrictions are in direct opposition to the required behaviour.

What can we learn from the present findings for understanding the role of linear synergy in human behaviour? Here conclusions must be drawn carefully. Neither the **SB** networks nor the **UC** networks could be observed to increase linear synergy as performance increases. Thus, linear synergy does not seem to be a priori a good strategy to master a pointing task. The advantage that imposed linear synergy means for evolutionary search nonetheless suggests that it may act as a useful constraint during development. This result relates well with the findings reported by Zaal et Al. [15] about infants employing linear synergy regardless of movement success. The authors hypothesise that linear synergy eases the acquisition of reaching and pointing movements at an early stage, a hypothesis supported by the presented results. In order to further investigate this hypothesis, it would be interesting to study the phylogeny of linear synergy, or, as an extension to the experiments presented here, to evolve the constraints for ontogenetic development. It would be desirable to abolish the restriction of hand movement to the plane in further experiments and include gravity in the model. These simplifications ease the task, but, at the same time lead to loss of biological plausibility. Extending the model this way would require the networks to additionally solve the non–trivial task of equilibrating forces involved.

Even though originating from a homuncular view of motor control, which is generally rejected by dynamical and artificial life perspectives, the concept of

motor synergies is fruitful for the investigation of motor behaviour. Within an evolutionary robotics framework, systematicities between effectors, as they are ubiquitous in humans and animals, can be investigated to find out about functions they may bear. Furthermore, as robots and tasks become more complex, description and explanation of the behaviour obtained become more complex as well. Looking for systematicities between effectors can be a good starting point in trying to understand intelligent behaviour, and designing experiments informed by observations on biological organisms may help to generate it.

References

1. Arbib, M. A.:*Perceptual Structures and Distributed Motor Control.* In: V. B. Brooks (ed.): *Handbook of Physiology. Section 2: The Nervous System.* Vol. II, Motor Control, Part 1. American Physiological Society (1981) 1449–1480.
2. Beer, R. D.: *Toward the Evolution of Dynamical Neural Networks for Minimally Cognitive Behavior.* In: P. Maes, M. J. Mataric, J.-A. Meyer, J. B. Pollack & S. W. Wilson (eds.): *From Animals to Animats 4. Proc. 4th Int. Conf. on Simulation of Adaptive Behavior.* Cambridge, MA: MIT Press 1996. 421–429.
3. Bernstein, N.: *The Coordination and Regulation of Movements.* Oxford: Pergamon 1967.
4. Gottlieb, G. L., Q. Song, G. L. Almeida, D. Hong, and D. Corcos: *Directional Control of Planar Human Arm Movement.* Journal of Neurophysiology 78 (1997) 2985–2998.
5. Grossberg, S. and Paine, R.W.: *A Neural Model of Corticocerebellar Interactions During Attentive Imitation and Predictive Learning of Sequential Handwriting Movements.* Neural Networks, 13 (2000) 999–1046.
6. Hebb, D.O.: *The Organization of Behavior. A Neuropsychological Theory.* John Wiley & sons inc., New York 1949.
7. Kandel, Eric R., James H. Schwartz and Thomas M. Jessel (eds.): *Principles of Neural Science.* Fourth Edition. McGraw–Hill, New York 2000.
8. Morasso, P., F.A. Mussa Ivaldi and C. Ruggiero: *How a Discontinuous Mechanism Can Produce Continuous Patterns in Trajectory Formation and Handwriting.* Acta Psychologica 54 (1983) 83–98.
9. Nolfi, S and D. Floreano: *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self–Organizing Machines.* MIT Press, Cambridge MA 2000.
10. Sporns, O., and G.M. Edelman: *Solving Bernstein's Problem: A Proposal for the Development of Coordinated Movement by Selection.* Child Dev. 64 (1993) 960–981.
11. Smith, R.: *Open Dynamics Engine. 0.5 release.* http://ode.org, retrieved: 10.1.2005.
12. Tuller, B., H. Fitch and M. Turvey: *The Bernstein Perspective: II. The Concept of Muscle Linkage or Coordinative Structure.* in: S. Kelso (ed.): *Human Motor Behavior. An Introduction.* Hillsdale: Lawrence Erlbaum (1982) 239–252.
13. Turvey, M., H. Fitch and B. Tuller: *The Bernstein Perspective: I. The Problems of Degrees of Freedom and Context-Conditioned Variability.* in: S. Kelso (ed.): *Human Motor Behavior. An Introduction.* Hillsdale: Lawrence Erlbaum (1982) 253–270.
14. Weiss, P. and M. Jeannerod: Getting a Grasp on Coordination. News Physiol. Sci. 13 (1998) 70–75.
15. Zaal, F., Daigle, K., Gottlieb, G.L., Thelen, E.: *An Unlearned Principle for Controlling Natural Movements.* Journal of Neurophysiology, 82 (1999) 255–259.

Self-assembly on Demand in a Group of Physical Autonomous Mobile Robots Navigating Rough Terrain

Rehan O'Grady[1], Roderich Groß[1], Francesco Mondada[2], Michael Bonani[2], and Marco Dorigo[1]

[1] IRIDIA, Université Libre de Bruxelles, Brussels, Belgium
{rogrady, rgross, mdorigo}@ulb.ac.be
[2] ASL, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
{francesco.mondada, michael.bonani}@epfl.ch

Abstract. Consider a group of autonomous, mobile robots with the ability to physically connect to one another (self-assemble). The group is said to exhibit *functional self-assembly* if the robots can choose to self-assemble in response to the demands of their task and environment [15]. We present the first robotic controller capable of functional self-assembly implemented on a real robotic platform.

The task we consider requires a group of robots to navigate over an area of unknown terrain towards a target light source. If possible, the robots should navigate to the target independently. If, however, the terrain proves too difficult for a single robot, the robots should self-assemble into a larger group entity and collectively navigate to the target.

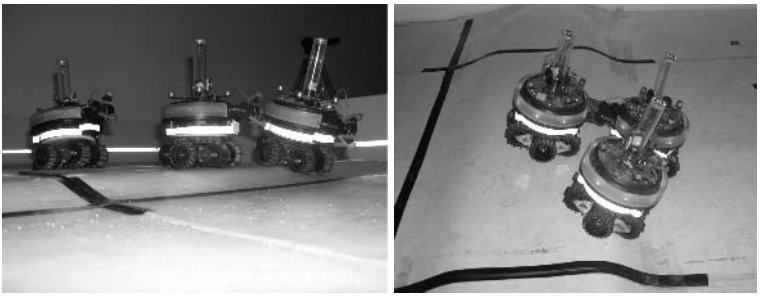
We believe this to be one of the most complex tasks carried out to date by a team of physical autonomous robots. We present quantitative results confirming the efficacy of our controller. This puts our robotic system at the cutting edge of autonomous mobile multi-robot research.

1 Introduction

Collective robotics addresses the design, implementation and study of multi-robotic systems. Swarm robotics is a subset of collective robotics which takes inspiration from social insect behaviour and emphasises *swarm intelligence* [2] principles such as decentralisation of control and use of local information. Many swarm robotics applications require cooperation between robots [8]. Some applications further require physical connectivity between cooperating robots. It is this last class of application that interests us. Although there is a large body of work on the capabilities of physically connected systems, very little research has been conducted on the mechanisms of when and how autonomous mobile agents should self-assemble.

The phrase *functional self-assembly* [15] describes a key adaptive response mechanism of distributed systems. We define self-assembly as the process

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 272–281, 2005.



(a) (b)

Fig. 1. (a) *S-bots* overcome a 2 cm hill independently. (b) *S-bots* self-assemble in order to overcome a 5 cm hill collectively.

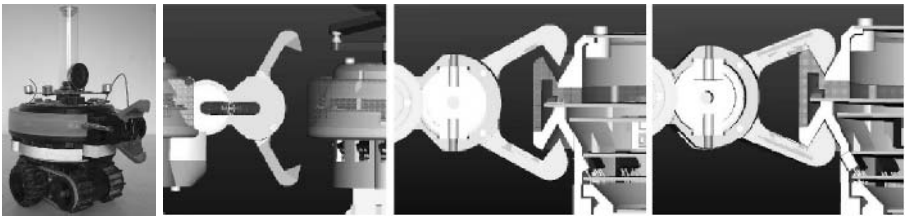
through which separate autonomous agents form a larger group entity by physically connecting to one another. If the agents can autonomously choose to self-assemble in response to the demands of their task and environment, they are said to display *functional self-assembly*.

A number of social insect species depend on functional self-assembly (for a review see [1]). Members of the ant species *Œcophylla longinoda*, for example, connect to one another to form bridges that other ants can then traverse [7]. Given its ubiquity in natural systems, functional self-assembly has been given surprisingly little attention by the swarm robotics community. In the only dedicated work, Trianni et al. [15] evolved neural network controllers for robots that needed to self-assemble and disassemble in order to traverse artificially designated 'hot' and 'cold' zones in a simple simulation environment.

Over the last decade, much of the research involving systems of physically connected robotic modules has been targeted at collective rough terrain navigation. In Hirose *et al.*'s system [6] modules are mechanically linked by means of a passive arm and are therefore incapable of self-assembly. Yim *et al.*'s system [16] can climb near vertical walls. Individual modules are incapable of autonomous motion and have very few external sensors for perception of the environment. Similar limitations are found in the majority of self-reconfigurable robotic systems, usually rendering self-assembly difficult or impossible [12,14].

In this paper we present the first physical robot controller capable of functional self-assembly. Our controller was implemented on the SWARM-BOT robotic platform [11,10,3]. This innovative system consists of a number of autonomous robotic agents called *s-bots*. *S-bots* are able to physically connect to one another, thus forming a larger group entity termed a *swarm-bot*. A *swarm-bot* can complete tasks impossible for a single *s-bot*. It can, for example, cross chasms wider than an *s-bot* or overcome hills too steep for a single *s-bot*.

The task we investigate requires a group of *s-bots* to navigate towards a target light source over unknown terrain. The *s-bots* must 'decide' whether or not to self-assemble based on the terrain they encounter. We use two different environments in our experiments. The first environment contains a simple hill which a single *s-bot* can overcome (see Fig. 1a). The *s-bots* can thus reach the



(a) (b)

Fig. 2. (a) The *s-bot*. (b) The *s-bot* gripping mechanism.

target independently. The second environment contains a steep hill too difficult for a single *s-bot*. The *s-bots* must self-assemble in order to overcome the hill and reach the target (see Fig. 1b).

2 Experimental Setup

2.1 The S-Bot

This study was conducted on the SWARM-BOT robotic platform [11,10,3]. The system consists of a number of mobile autonomous robots called *s-bots* (see Fig. 2a). The *s-bot* is equipped with a traction system made up of tracks and wheels. This chassis provides the *s-bot* with efficient on the spot rotation and mobility on moderately rough terrain. The majority of the *s-bot* sensory and processing systems are housed in a turret mounted above the chassis. A motorised axis allows this turret to rotate with respect to the chassis.

Physical connections between two *s-bots* can be established by a gripper-based connection mechanism (see Fig. 2b). Each *s-bot* is surrounded by a T-shaped ring which can be grasped by other *s-bots*.

The *s-bot* sensory systems used in this study are as follows: 15 proximity sensors distributed around the turret allow for the detection of obstacles. A 3-axes accelerometer provides information on the *s-bots*' inclination which can be used to detect if the *s-bot* is in danger of falling. The connection ring of the *s-bot* is equipped with eight groups of coloured LEDs. An omni-directional camera is mounted on the turret. The combination of the camera and the LED ring allows an *s-bot* to communicate its presence and even its internal state to other nearby *s-bots*. Inside the gripper is an optical light barrier to detect the presence of objects to be grasped. Other sensors provide the *s-bot* with information about its internal motors. This includes positional information (e.g., of the rotating turret) and torque information (e.g., of forces acting on the tracks).

2.2 The Task

We conduct experiments in two different environments (see Fig. 3). Both measure 240 cm x 120 cm and consist of two areas of flat terrain (a starting area and a target area) separated by an area of rough terrain. In *Environment A*, the rough

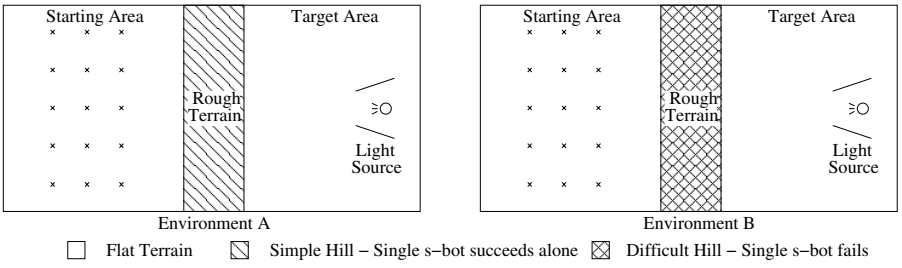


Fig. 3. Scale diagram of the two experimental environments (view from above). *S-bot* starting positions are marked by crosses.

terrain is a 2 cm high hill which can be overcome by a single *s-bot* (see Fig. 1a). In *Environment B* the rough terrain hill is 5 cm high - too difficult for a single *s-bot* (see Fig. 1b).

The initial position of each *s-bot* in the starting area is assigned randomly by uniformly sampling without replacement from a set of 15 possible starting points. The *s-bot*'s initial orientation is chosen randomly from a set of 4 possible directions. To complete the task the *s-bots* must reach the target area without toppling over.

The *s-bots* have no a priori knowledge of the environment they are in—they must react to the environment and determine whether or not to self-assemble. In *Environment A* the *s-bots* should navigate to the target area independently. In *Environment B* the *s-bots* must aggregate, self-assemble and collectively overcome the hill in order to reach the target area.

3 Controller

We use a distributed behaviour-based controller (see Fig. 4). Each *s-bot* is fully autonomous. The same controller is executed on every *s-bot*. An *s-bot* starts by navigating independently towards the target light source. If the *s-bot* finds a hill too difficult for it to pass alone, or if it sees another *s-bot* that is either aggregating or assembled (sees blue or red), it illuminates its blue LEDs and starts aggregating. An aggregating *s-bot* can probabilistically trigger self-assembly by

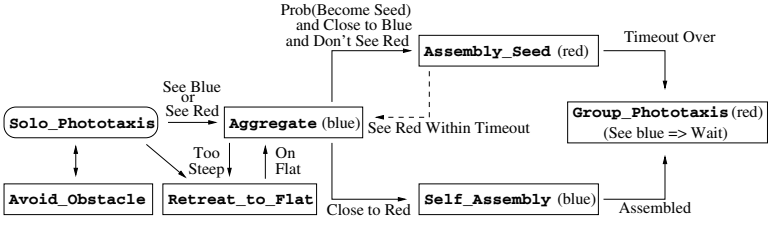


Fig. 4. Behaviour transition model for the behaviour-based controller

`Aggregate` behaviour	`Self_Assembly` behaviour

loop
  if canSeeClose(red) **then**
  switchBehaviour(`Self_Assembly`)
  else if canSeeFar(red) **then**
  approachRed( )
  else if canSeeClose(blue) **then**
  $Prob(0.04)$
  $\rightarrow$ switchBehaviour(`Assembly_Seed`)
  $Prob(0.96) \rightarrow$ doNothing( )
  else if canSeeFar(Blue) **then**
  approachBlue( )
  else
  randomWalk( )
  end if
end loop
	repeat
	  $(i_1, i_2) \leftarrow$ featureExtraction(camera)
	  $(i_3, i_4) \leftarrow$ sensorReadings(proximity)
	  $(o_1, o_2, o_3) \leftarrow f(i_1, i_2, i_3, i_4)$
	  if graspingRequirementsMet(o_3) **then**
	  try to grasp
	  else
	  applyValuesToTracks(o_1, o_2)
	  end if
	until successfully connected

Fig. 5. Algorithms for `Aggregate` behaviour (left) and `Self_Assembly` behaviour (right)

illuminating its red LEDs and becoming a static seed. Aggregating *s-bots* assemble to the seed *s-bot* or to already assembled *s-bots* (any red object). Assembled *s-bots* illuminate their red LEDs then perform group phototaxis once they can no longer detect any unassembled *s-bots* (can no longer see blue).

- `Solo_Phototaxis`. This is the starting behaviour. The *s-bot* uses its camera to navigate towards the target light source. The *s-bot* uses its accelerometers to reduce maximum track speed as a linear function of inclination. This is to prevent the *s-bot* toppling before `Retreat_to_Flat` behaviour is triggered.
- `Avoid_Obstacle`. This behaviour is triggered when the readings from the *s-bot*'s 15 proximity sensors exceed a threshold. The *s-bot* determines the direction of the obstacle using its proximity sensors then moves in the opposite direction until the proximity threshold is no longer exceeded.
- `Retreat_to_Flat`. This behaviour is initiated when the *s-bot*'s accelerometers indicate that the *s-bot* is in danger of toppling over. The *s-bot* reverses downhill to flat terrain, reverses away from the rough terrain, then rotates to face away from the slope.
- `Aggregate`. This behaviour is detailed in Fig. 5 (left). The *s-bots* must locate and then approach each other as a precondition for self-assembly. Values for the hard coded probabilities were manually optimised through trial and error.
- `Self_Assembly`. This behaviour is detailed in Fig. 5 (right). Function f maps sensory input (i_1, i_2, i_3, i_4) to motor commands (o_1, o_2, o_3). It is implemented by a neural network which was designed by artificial evolution and tested with physical robots in previous works [5,4].
- `Assembly_Seed`. This behaviour is necessary to trigger the self-assembly process. If a red object is detected within 3 s of behaviour initiation, control is passed to `Aggregate` behaviour. (This prevents multiple seeding—if two nearby *s-bots* switch to `Assembly_Seed` behaviour, both will revert to `Aggregate` behaviour). After 3 s control is passed to `Group_Phototaxis` behaviour.

- `Group_Phototaxis`. The *s-bot* remains stationary if it detects blue objects in the vicinity (*s-bots* still assembling). Otherwise the *s-bot* performs phototaxis to the target. Because it is part of a *swarm-bot*, the orientation of the turret is fixed. The *s-bot* continually rotates the traction system with respect to the turret to keep the tracks oriented towards the target [5].

4 Results

We conducted a series of experiments in two different environments (see Fig. 3) with groups of 1, 2 and 3 *s-bots*.[1]

Trials with 3 *s-bots* in Environment A. We conducted 20 trials. In every trial all 3 *s-bots* reached the target zone. In 19 out the 20 trials the *s-bots* correctly navigated independently to the target. In a single trial the *s-bots* self-assembled on the down slope of the hill and then performed collective phototaxis to the target. The incorrect decision to self-assemble was due to a colour misperception of a non-existent object by an *s-bot*.

Trials with a single *s-bot* in Environment B. We modified the controller to only execute `Solo_Phototaxis` behaviour. The *s-bot* was thus limited to navigating towards the target taking no account of the terrain encountered.

We conducted 20 trials. The *s-bot* failed to overcome the hill in 20 out of 20 trials. In each trial the *s-bot* reached the hill and then toppled backwards due to the steepness of the slope.

To confirm that the *s-bot* was failing due to the intrinsic properties of the slope, we repeated this experiment at a number of different constant speeds.

Trials with 2 *s-bots* in Environment B. We conducted 20 trials. The *s-bots* successfully detected the slope in every trial. Furthermore the *s-bots* always succeeded in assembling into a 2 *s-bot swarm-bot*. In 13 trials (65%) the *swarm-bot* succeeded in overcoming the hill. In the other 7 trials (35%) the assembled *swarm-bot* failed to overcome the hill. These failures happened when the assembled *s-bots* attempted to climb the hill in parallel.

Trials with 3 *s-bots* in Environment B. We conducted 20 trials. The *s-bots* successfully detected the slope in every trial. In 16 trials (80%) all of the *s-bots* successfully self-assembled into a 3 *s-bot swarm-bot*. In each of these 16 trials the 3 *s-bot swarm-bot* went on to successfully reach the target area. Fig. 6 shows a sequence of images from a typical trial.

In the remaining 4 trials (20%) the *s-bots* still managed in each case to self-assemble into a *swarm-bot* of 2 *s-bots*. In two of these 4 trials the *swarm-bot* went on to successfully reach the target area. In the two other trials the *swarm-bot* was obstructed by the third *s-bot* which failed to self-assemble.

[1] Videos of all experiments can be found at `http://iridia.ulb.ac.be/~rogrady/ecal2005/`

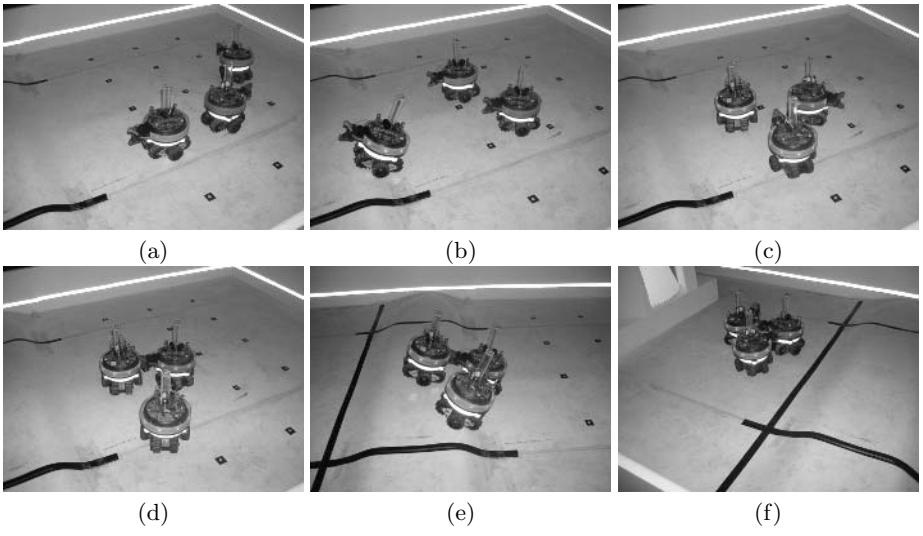


Fig. 6. The *s-bots* start in a random configuration (a). One *s-bot* detects a slope it cannot overcome alone and activates blue LEDs (b). The other *s-bots* detect blue colour (local communication). The group aggregates and self-assembles (c,d). The *s-bots* collectively overcome the rough terrain and reach the target area (e,f).

Table 1. Percentage of *s-bots* in Environment B trials succeeding for Self-assembly (**A**) and Completion of task (**C**)

	1 *s-bot* trials		2 *s-bot* trials		3 *s-bot* trials	
	A	**C**	**A**	**C**	**A**	**C**
% Successful (total)	-	0.00	100.00	65.00	93.33	86.67
% Successful alone	-	0.00	-	0.00	-	0.00
% Successful in 2 *s-bot swarm-bot*	-	-	100.00	65.00	13.33	6.67
% Successful in 3 *s-bot swarm-bot*	-	-	-	-	80.00	80.00
% Failed	-	100.00	0.00	35.00	6.67	13.33

4.1 Analysis

Table 1 shows the percentage of *s-bots* that successfully self-assembled and the percentage of *s-bots* that successfully completed the entire task in the Environment B experiments. The three columns distinguish between trials with 1 *s-bot*, 2 *s-bots* and 3 *s-bots*. The first row shows the total percentage of successful *s-bots*. Subsequent rows show the percentage of *s-bots* that were successful alone, or as part of a 2 *s-bot swarm-bot* or as as part of a 3 *s-bot swarm-bot*, or that failed.

The success rate for task completion increases with the number of robots. A single robot always fails. In 2 *s-bot* trials, 65% of *s-bots* complete the task. The 3 *s-bot* trials show a further clear improvement—86.67% complete the task.

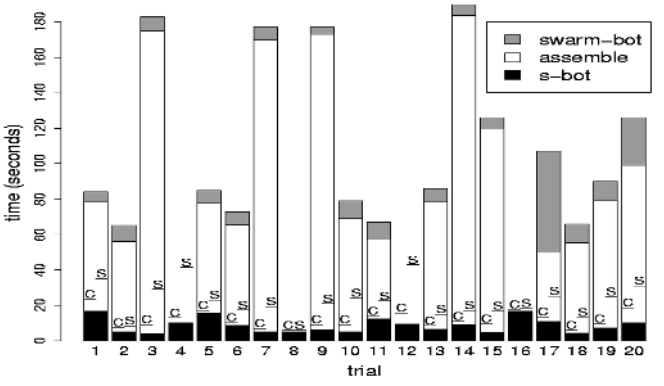


Fig. 7. 3 *s-bot* trials in Environment B. Phases represented are: (i) s-bot: independent *s-bot* navigation; (ii) assembly: aggregation and self-assembly; (iii) swarm-bot: collective *swarm-bot* navigation.

The fourth row (% Successful in 3 *s-bot swarm-bot*) shows that in the 3-*s-bot* trials 80% of *s-bots* successfully self-assemble into a 3 *s-bot swarm-bot*. The same row shows us that 80% of *s-bots* complete the task in a 3 *s-bot swarm-bot*. Thus in 3 *s-bot* trials, whenever all the 3 *s-bots* successfully self-assemble into a 3 *s-bot swarm-bot* they always successfully overcome the rough terrain. By contrast, in the 2 *s-bot* trials 100% of the *s-bots* self-assemble into a 2 *s-bot swarm-bot*. Despite this, only 65% of the 2 *s-bot swarm-bots* successfully overcome the hill.

The hill in environment B is such that in our trials a 3 *s-bot swarm-bot* always (100% of the trials) overcomes it. A 2 *s-bot swarm-bot* on the other hand sometimes (35% of the trials) fails to overcome the hill. Whenever the 2 *s-bot swarm-bot* approached the hill in parallel the *swarm-bot* toppled backwards.

Fig. 7 illustrates three phases of task completion. In the first phase (black segment) all *s-bots* are independently navigating to the target (this phase ends when the hill is first detected by an *s-bot*). The phase takes between 4 s and 17 s depending on the random initial configuration of the *s-bots*. For the unsuccessful trials (4,8,12,16) only this first phase is illustrated.

The second phase (white segment) consists of aggregation and self-assembly. This phase takes between 39 s and 175 s. This phase always accounts for a large percentage of total completion time due to its high level of complexity.

The final phase (grey segment) consists of collective phototaxis to the target. This phase takes between 4 s and 30 s, except in trial 17, when the *swarm-bot* got stuck for some time on the hill.

The symbol 'c' in Fig. 7 marks the first time that all *s-bots* become aware of the hill. In some trials (e.g. trials 5 and 6) the existence of the difficult hill is communicated very quickly between *s-bots* (see also Fig. 6). One *s-bot* detects the rough terrain and activates its blue ring LEDs. The other *s-bots* are already close enough to detect this blue colour. In such trials the point 'c' is reached soon after the start of the aggregation and self-assembly phase. In other trials

(e.g. trials 1 and 12) it takes longer to reach point 'c' as the *s-bots* are sufficiently far apart that two *s-bots* discover the hill independently.

The symbol 's' in Fig. 7 indicates when self-assembly was seeded (the last time an *s-bot* switches to `Assembly_Seed` behaviour).

5 Conclusion

Self-assembly is a critical adaptive response mechanism in a number of social insect species. This work represents the first successful use of this response mechanism by real robots. We have shown that a group of physical autonomous mobile robots can choose to self-assemble in response to the demands of their task and environment. Using our controller, a group of robots faced with a simple hill overcome it independently. When the same robots are faced with a hill too difficult for a single robot they self-assemble and overcome the hill together. The success rate increased with the number of robots used: 0%,65% and 86.67% for groups of 1, 2 and 3 robots respectively.

Our approach involved splitting the task (as seen from the perspective of an individual robot) into distinct phases. Each phase was addressed by a separate behaviour module - these modules were combined to produce our behaviour based controller. In a previous work conducted in a simplified simulation environment, Trianni et al. [15] focused on evolving a single neural network controller to achieve functional self-assembly. We believe that application of this evolutionary approach to the real robots might yield solutions that exploit hidden properties of the robotic hardware or which make better use of the complex group dynamics of the task [13].

We are currently investigating mechanisms to generate connection patterns and group sizes [9] that are suited to particular tasks. In the spirit of functional self-assembly we would like the robots themselves to choose these patterns and group sizes as they interact with their environment.

Acknowledgements

This work was supported by the ANTS project, an *Action de Recherche Concertée* funded by the Scientific Research Directorate of the French Community of Belgium; by the SWARM-BOTS project, funded by the Future and Emerging Technologies programme of the European Commission (grant IST-2000-31010); and by COMP2SYS, a Marie Curie Early Stage Research Training Site funded by the European Community's Sixth Framework Programme (grant MEST-CT-2004-505079). The information provided is the sole responsibility of the authors. It does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data in this publication. Marco Dorigo acknowledges support from the Belgian FNRS, of which he is a Research Director.

References

1. C. Anderson, G. Theraulaz, and J.-L. Deneubourg. Self-assemblages in insects societies. *Insectes Soc.*, 49:99–110, 2002.
2. E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York, NY, 1999.
3. M. Dorigo, V. Trianni, E. Şahin, R. Groß, T. H. Labella, G. Baldassarre, S. Nolfi, J.-L. Deneubourg, F. Mondada, D. Floreano, and L. M. Gambardella. Evolving self-organizing behaviors for a Swarm-bot. *Auton. Robots*, 17(2–3):223–245, 2004.
4. R. Groß, M. Bonani, F. Mondada, and M. Dorigo. Autonomous self-assembly in mobile robotics. Technical Report IRIDIA/2005-2, IRIDIA - Université Libre de Bruxelles, 2005. Submitted to *IEEE Trans. Robot.*
5. R. Groß and M. Dorigo. Group transport of an object to a target that only some group members may sense. In *Parallel Problem Solving from Nature – 8th Int. Conf. (PPSN VIII)*, volume 3242 of *Lecture Notes in Computer Science*, pages 852–861. Springer Verlag, Berlin, Germany, 2004.
6. S. Hirose, T. Shirasu, and E. F. Fukushima. Proposal for cooperative robot "Gunryu" composed of autonomous segments. *Robot. Auton. Syst.*, 17:107–118, 1996.
7. A. Lioni, C. Sauwens, G. Theraulaz, and J.-L. Deneubourg. Chain formation in *Œcophylla longinoda*. *J. Insect Behav.*, 15:679–696, 2001.
8. A. Martinoli, K. Easton, and W. Agassounon. Modeling swarm robotic systems: A case study in collaborative distributed manipulation. *Int. J. Robot. Res.*, 23(4):415–436, 2004.
9. C. Melhuish, O. Holland, and S. Hoddell. Convoying: Using chorusing to form travelling groups of minimal agents. *Robot. Auton. Syst.*, 28:207–216, 1999.
10. F. Mondada, L. M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneubourg, and M. Dorigo. SWARM-BOTS: Physical interactions in collective robotics. *IEEE Robot. Autom. Mag.*, 2005, to appear.
11. F. Mondada, G. C. Pettinaro, A. Guignard, I. V. Kwee, D. Floreano, J.-L. Deneubourg, S. Nolfi, L. M. Gambardella, and M. Dorigo. SWARM-BOT: A new distributed robotic concept. *Auton. Robots*, 17(2–3):193–221, 2004.
12. S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji. M-TRAN: Self-reconfigurable modular robotic system. *IEEE/ASME Trans. Mechatron.*, 7(4):431–441, 2002.
13. S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, Cambridge, MA, 2000.
14. M. Rubenstein, K. Payne, P. Will, and W. M. Shen. Docking among independent and autonomous CONRO self-reconfigurable robots. In *Proc. of the 2004 IEEE Int. Conf. on Robotics and Automation*, volume 3, pages 2877–2882. IEEE Computer Society Press, Los Alamitos, CA, 2004.
15. V. Trianni, E. Tuci, and M. Dorigo. Evolving functional self-assembling in a swarm of autonomous robots. In *From Animals to Animats VIII. Proc. of the* 8^{th} *Inter. Conf. on Simulation of Adaptive Behavior*, pages 405–414. MIT Press, Cambridge, MA, 2004.
16. M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw, and S. B. Homans. Modular reconfigurable robots in space applications. *Auton. Robots*, 14(2-3):225–237, 2003.

Superlinear Physical Performances in a SWARM-BOT

Francesco Mondada, Michael Bonani, André Guignard, Stéphane Magnenat, Christian Studer, and Dario Floreano

Laboratory of Intelligent Systems,
Ecole Polytechnique Fédérale de Lausanne (EPFL)
http://lis.epfl.ch, http://www.swarm-bots.org

Abstract. A *swarm-bot* is a robotic entity built of several autonomous mobile robots (called *s-bots*) physically connected together. This form of collective robotics exploits robot interactions both at the behavioral and physical levels. The goal of this paper is to analyze the physical performance of a swarm-bot as function of its size (number n of s-bots composing it). We present three tasks and the corresponding swarm-bot performances. In all three tasks we show superlinear performances in a range of n where the physical forces applied in the structure fit to the robot design. This superlinear performance range helps in understanding which swarm-bot size is optimal for a given task and gives interesting hints for the design of new application-oriented swarm-bots.

1 Introduction

Swarm-bot is a new robotic concept [11] that takes inspiration from insect self-assembling capabilities. For instance some ants use their legs and mandibles to connect to each other in order to form structures such as bridges or rafts [1]. Similarly, in a swarm-bot the cooperation among single mobile robots (called *s-bots*) is achieved by physical connections [10] (see figure 1). This approach generates new physical properties such as robustness, flexibility and, in some cases, improved physical performances. The **robustness** of this type of distributed system has been well studied in collective robotics [12,3,2] and is made possible by the redundancy of the system. The **flexibility** of a swarm-bot is given by its modularity and self-assembling ability. Self-reconfigurable robots show similar properties in their modularity and are also well studied [6,5].

This paper focuses on the physical **performance** of this new type of self-assembling robotic system. A simplistic way of showing collective performances consist in measuring *threshold performances* on strictly collective[1] tasks, where a single robot cannot solve the problem alone and need the help of other robots to achieve the task. In this type of task the performance can be expressed by the number of robots (threshold m) necessary to solve the task. Even if this threshold does not represent a real quantification of the performances, it is sufficient

[1] *Strictly* collective tasks need the collaboration of more than one individual. *Loosely* collective tasks can be solved by one individual having sufficient time [8].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 282–291, 2005.

to demonstrate that the task is strictly collective and to show the ratio between individual and collective performances. Typical examples of strictly collective tasks are object pushing [9] or lifting [4]. In swarm robotics there are similar examples for gap or step passing [11,10]. It is more difficult but also more interesting to collect *quantitative performances*, both on strictly or loosely collective tasks. This type of measurement can give a better indication of the improvement the collective system brings as function, for instance, of the number of robots used. An interesting characteristic of collective performances is the collective speedup factor [8] of a group of n robots, given by equation 1.

$$CS(n) = \frac{mP(n)}{nP(m)} \tag{1}$$

where $P(n)$ is the performance of a group of n robots and m is the minimal number of robots needed to perform the task. We can distinguish between superlinear performances when $CS(n) > 1$, linear performances when $CS(n) = 1$ and sublinear performances when $CS(n) < 1$. A simple combination of n robots having no influence on each-other should generate a linear performance by performing the task n times better or faster than one robot or module (see for instance gap or step passing with polypod [13]). In most situations it is hard to avoid interferences between the robots, for instance because of a common resource. Those interferences often affect performance and generate sublinear properties (see for instance a simple object clustering using Khepera [7]). In some situations, however, interferences are constructive and help in better solving the task, generating superlinear performances (see the collaboration rate in a stick pulling task [4] or the pushing time for two robots pushing a box [9]). Tasks where we can observe superlinear performances are of course the best application area for collective robotics.

In addition to the characterization of the linearity of the performance, it is important to verify the scalability of this property. It is of course more interesting when these properties scale well to a high number of robots.

Fig. 1. A swarm-bot robot composed by six *s-bots* entering a building from a narrow passage (left) and passing a large gap (right)

Fig. 2. Left: Chain of five s-bots forming a swarm-bot and pulling on a dynamometer. Right: One s-bot pulling the same dynamometer.

This paper focus on swarm-bot performances based on three experiments: Object pulling, gap passing and step passing. The next three sections show that a swarm-bot can solve these task with superlinear performances.

2 Object Pulling

2.1 Setup

The goal is to pull an object using a swarm-bot in chain configuration, as illustrated in figure 2 left. To measure the performances of the swarm-bot, the pulling force is measured by a dynamometer connected to the first s-bot. The other s-bots form a chain behind the first one, pulling in the same direction. The robots are remotely controlled by an operator. The pulling force is measured as function of the number of robots.

2.2 Results and Discussion

Table 1 shows the average pulling force (over three tests) of swarm-bot of different sizes (n) and on four different ground conditions. The collective speedup $CS(n)$ is given for each ground condition and averaged at the end.

Table 1. Pulling force

Number of s-bots n composing the swarm-bot	1	2	3	4	5
Average pulling force ground 1 $P_1(n)$ [N]	2.65	6.75	11.4	15.1	18.5
$CS_1(n)$	1	1.27	1.43	1.42	1.4
Average pulling force ground 2 $P_2(n)$ [N]	2	8	11	12	17.5
$CS_2(n)$	1	2	1.8	1.5	1.75
Average pulling force ground 3 $P_3(n)$ [N]	3	7.5	12	13.5	15
$CS_3(n)$	1	1.25	1.33	1.12	1
Average pulling force ground 4 $P_4(n)$ [N]	4.7	10	11.6	19.2	23.5
$CS_4(n)$	1	1.06	0.82	1.02	1
Average CS(n)	1	1.4	1.36	1.27	1.29

Table 2. Gap passing performance

		Number of robots				
		1	2	3	4	5
Polypod earthworm	$P(n)$ = gap V_d [cm]	2.8	5.6	8.4	11.2	14
	$CS(n) = \frac{P(n)}{nP(1)}$	1	1	1	1	1
Swarm-bot	$P(n)$ = gap size [cm]	4	9	18	22	22
	$CS(n) = \frac{P(n)}{nP(1)}$	1	1.125	1.5	1.375	1.1

Table 1 shows that in average a swarm-bot composed by two s-bots displays superlinear performance in respect to a single s-bot. Only on ground number four the performance is nearly linear. In average two robots together perform 1.4 times better than the sum of their individual performances. The collective speedup shows superlinear performances up to five robots, but for $n > 2$ the speedup is less important. The large performance step between one and two s-bots is generated by a physical stabilization of the pulling structure. As shown on the right of figure 2, an s-bot alone cannot generate an optimal pulling force because of the position of the gripper in respect to the tracks and the center of mass, resulting in the s-bot pulling only with the front part of the tracks.

When two s-bots ($n = 2$) build together a swarm-bot structure, they achieve a much better stability. In this structure the center of mass is better placed with respect to the structure and both s-bots can pull with tracks and wheels well placed on the ground. This allows each robot to provide a maximal performance, much better than the performance they would achieve alone. For a swarm-bot composed by more than two s-bots there is no additional structural improvement, which is shown by the decreasing value of $CS(n)$ for $n > 2$. This decreasing collective speedup is also a result of the loss of forces in the chain when $n > 3$, not allowing a proper addition of the individual performances. This loss of performances is due to the connections, the orientation of the robots and the inter-robot efforts. For a larger number of robots ($n > 5$) forces are sufficiently strong to break the chain or even break the gripper of the first robot of the chain. For this reason tests have been made only for $n \leq 5$.

3 Passing a Gap

3.1 Setup

In this task the goal is to pass a gap. The swarm-bot configuration is a chain as in the previous experiment (see figure 3) but we exploit here the rigidity of the chain. In this experiment the robots are assembled by hand and controlled by a simple program performing hole detection and robot lifting to compensate structure bending. Depending on the gap size, this is a strictly collaborative task with a threshold performance. We have quantified the performance in respect to the number of s-bot by measuring the maximal gap size the swarm-bot structure can pass. This parameter was introduced by Yim [13] for the polypod performances, under the term of V_d.

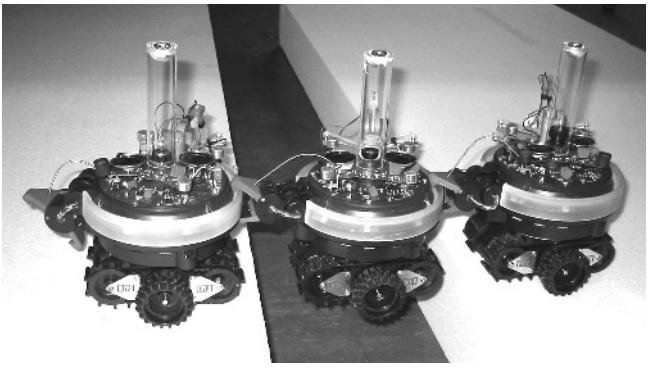


Fig. 3. Three s-bots passing a gap in swarm-bot configuration

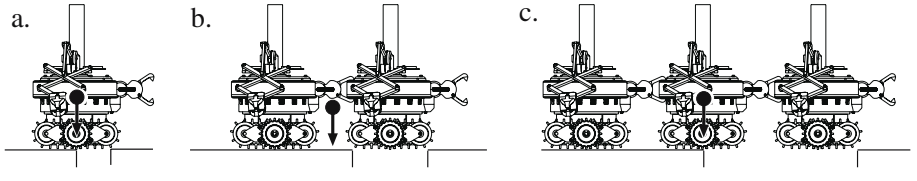


Fig. 4. Center of mass of a swarm-bot facing a gap, depending on the number of s-bot connected. The maximal gap size the swarm-bot can pass depends on the position of the center of mass in respect to the tracks.

3.2 Results and Discussion

Table 2 summarizes the performances given by Yim [13] for the polypod self-reconfigurable robot, the maximal gap size a swarm-bot composed by n s-bots can pass ($P(n)$) and the resulting collective speedups $CS(n)$. For $n \geq 4$ the gap size a swarm-bot can pass can be considered constant, because the gripper cannot support more than two s-bots suspended horizontally. Furthermore, when two s-bots are suspended horizontally, the third robot supporting them has a very strong pressure on the tracks. In the actual version, this pressure can block the tracks and immobilize the s-bot, stopping the whole swarm-bot.

Despite the limitations mentioned above for $n \geq 4$, the measurements show a distinctive collective speedup for n between 3 and 4. The best speedup is obtained for $n = 3$ which is explained by a structural reason, as illustrated in figure 4. When two s-bots self-assemble into a swarm-bot, their structure becomes longer than twice the length of an s-bot because of the length of the connecting device. This extra length is not well exploited for a swarm-bot composed by two s-bots because the center of mass is not placed over a supporting track. With three s-bots the swarm-bot can fully exploit its length because the center of mass is situated over the tracks of the second s-bot in the chain.

Table 2 shows clearly that the collective speedup observed on swarm-bots and on self-reconfigurable robots are very different. For swarm-bots the speedup is superlinear in a short range, corresponding to the limited capabilities of each s-bot[2]. In self-reconfigurable robots the speedup is linear but for a bigger range, corresponding to the bigger structure this type of robots can build.

4 Passing a Step

4.1 Setup

In this task, the goal of the swarm-bot is to pass a step. For a step size bigger than two centimeters this is a strictly collaborative task with a threshold performance, as for the gap passing task. Also in this case we use a chain configuration. A peculiarity of this task is that the swarm-bot must bend properly to pass the step, as illustrated in figure 5. To achieve this task the s-bots have been remotely controlled. We have quantified the performance with respect to the number of s-bots by measuring the maximal step size the swarm-bot can pass. This parameter was introduced by Yim [13] for the Polypod performance, under the term of V_b.

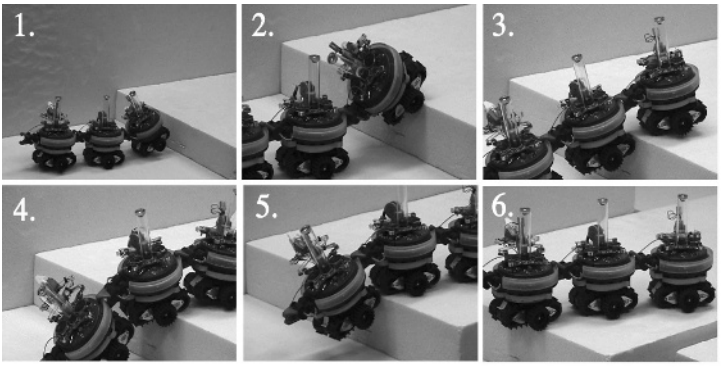


Fig. 5. Sequence of actions a swarm-bot composed by three s-bots must execute to pass a step of 10 cm

4.2 Results and Discussion

Table 3 summarizes the performances given by Yim [13] and those measured on swarm-bots. The number of s-bots has been limited to five because of mechanical constraints: The mechanical effort applied to the connection ring around the robot is very important when bending the swarm-bot structure (see [10] for details). With the actual hardware and for $n > 5$ the ring can broke. Because of this limitation, the step size ($P(n)$) would not increase significantly for $n > 5$.

[2] Each s-bot can lift, with its gripper, only one other s-bot. In self-reconfigurable robots a module can lift several other modules.

Table 3. Step passing performance

		Number of robots				
		1	2	3	4	5
Polypod earthworm	$P(n)$ = gap V_b [cm]	2.8	5.6	8.4	11.2	14
	$CS(n) = \frac{P(n)}{nP(1)}$	1	1	1	1	1
Swarm-bot	$P(n)$ = step size [cm]	1.5	4.5	10	14	16
	$CS(n) = \frac{P(n)}{nP(1)}$	1	1.5	2.22	2.33	2.13

This task shows impressive results from the point of view of all-terrain navigation. The s-bots can pass a step of their own size, which is a performance only few self-reconfigurable robots such as M-Tran [6] can achieve.

The collective speedup shown in this task is the highest among all experiments that we performed. The larger $CS(n)$ value is obtained for $n = 4$, but superlinear speedups are already obtained for $n = 2$ and continue for $n = 5$. The reasons of this superlinear performance are to be found in the better structural stability of the swarm-bot configuration. A single s-bot, despite its tracks, has a very limited all-terrain navigation capability, due to its relatively high center of mass. This is the reason of the very poor performance of one s-bot in step passing. A swarm-bot configuration made of two s-bots has a much better stability and can deal with a larger variety of terrain conditions. In addition to the stability of the structure, this task exploits the flexibility of the swarm-bot configuration. By bending its structure, the swarm-bot can improve significantly the all-terrain mobility. This is also the main explanation for the large $CS(n)$ value in the case of $n = 3, 4$ and 5. These configurations use more physical links and thus more flexibility in the chain shape, enabling better obstacle passing.

Also in this case the collective speedup shows superlinear performances which are very different from the linear performances shown in self-reconfigurable robots [13]. Even if the superlinearity range ($2 < n < 6$) was bigger than in the previous two experiments, it is smaller than the one observed for linear performances in self-reconfigurable robots (generally $n > 10$).

5 Conclusion

We presented three experiments showing swarm-bot performances as function of the number of s-bots composing it. We can observe two main properties:

1. **Superlinear performance**: All three experiments show superlinear performance. This is a clear indication that the physical connection plays a constructive role in the collaboration between robots. This constructive interaction between s-bots results in performances that are by far bigger than those obtained by the sum of the single robots contributions.

   Most of the superlinear performances generally observed in collective robotics are due to an optimal task distribution. This is not the case of the swarm-bot. In our experiments the superlinearity is due to a mechanical

improvement of the system and applies to physical tasks. This is a new phenomenon in collective robotics that deserves further exploration.

2. **Limited scalability**: The scalability of the results is limited to a small range ($2 < n < 5$) in swarm size. This is a clear limitation of our system, but the upper limits are clearly linked to physical and mechanical characteristics of the robot design. This means that the designer of the robot has an influence on these performances. Despite the possibilities of design improvement, physical limitations will always put an upper boundary to the superlinearity of this type of performances.

Although superlinear performances are a well known phenomenon in collective robotics, our experience brings a new element showing these properties in physical tasks using self-assembling capabilities at the robot level. This is a key aspect of our approach and can radically change the way of designing a robotic system for tasks such as all-terrain navigation or object transportation.

Global performance has shown to strongly depend on small design details. The detailed s-bot design choices for all-terrain navigation and inter-robot connection clearly shape the individual and swarm performances. This is a key issue in collective robotics engineering that has already been shown in other projects [4] and is shown here at the level of physical connection. Small implementation details have even more impact on the performance of the collective system if these performances are amplified in a superlinear matter. Our results should be an additional motivation to develop better design techniques to exploit collective speedup from the beginning of the design phase by predicting the performance of the collective system.

Figure 6 summarizes the collective speedup of the three experiments described in this article. We can see that the smallest speedup is obtained by the simplest task (object pulling) where the features of the physical link between the s-bots are less exploited. In this case the link is used only for creating a pulling connection, but does not exploit the rigidity or the mobility of the link. The second task uses the rigidity of the link and gets a better speedup with a maximal

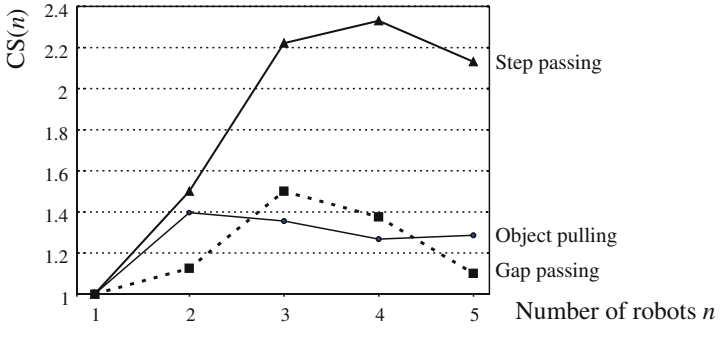


Fig. 6. Summary of the collective speedup as function of the number of s-bots and for each of the three tasks presented in this article

value for an higher number of robots. The last task, step passing, exploits all the properties of the physical link (rigidity and mobility) and achieves the best collective performances for a higher number of robots. These results demonstrate the relationship between exploitation of the physical link and collective speedup.

Another interesting point to observe is the difference between the performance of a swarm-bot and those of self-reconfigurable robots. Self-reconfigurable robots show nice linear performances mainly due to the simplicity and mechanical strength of their modules, allowing linear addition of the performances [13]. Our design uses as basic building block a fully autonomous individual with more complexity than a self-reconfigurable module, with much more weaknesses and relatively limited capabilities. This choice brings sublinear performances for high numbers of robots ($n > 10$) but superlinear ones for small swarm-bots ($n < 10$).

The experiments described above give an indication of the optimal size of a swarm-bot when addressing physical tasks. For example the performance measured shows that chains of four robots exploit in an optimal way this capability. Therefore the most efficient swarm-bots for all-terrain navigation should have a radius of four s-bots. These indications will be also useful for the design of new swarm-bots designed for specific applications.

Acknowledgment

Many thanks to Julien Pilet and René Beuchat from the LAP lab[3], to Vito Trianni, Antoine Beyeler and Michele Bongiovanni for their help and to all the SWARM-BOTS project partners for the inputs, collaboration and suggestions.

The SWARM-BOTS project is funded by the Future and Emerging Technologies programme (IST-FET) of the European Community, under grant IST-2000-31010. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication. The Swiss participants to the project are supported by the Swiss Government.

References

1. C. Anderson, G. Theraulau, and J.-L. Deneubourg. Self-assemblages in insect societies. *Insectes Sociaux*, 49:99–110, 2002.
2. Toshio Fukuda, Hiroshi Mizoguchi, Kaoru Sekiyama, and Fumihito Arai. Group Behavior Control for MARS (Micro Autonomous Robotic System). In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '99)*, pages 1550–1555. IEEE Press, Piscataway, New Jersey, USA, May 10th–15th 1999.
3. Dani Goldberg and Maya J. Matarić. Design and Evaluation of Robust Behavior-Based Controllers. In Tucker Balch and Lynne E. Parker, editors, *Robot Teams: From Diversity to Polymorphism*, pages 315–344. A. K. Peters, Natick, MA, USA, 2002.

[3] http://lapwww.epfl.ch

4. Auke Jan Ijspeert, Alcherio Martinoli, Aude Billard, and Luca Maria Gambardella. Collaboration through the exploitation of local interactions in autonomous collective robotics: the stick pulling experiment. *Autonomous Robots*, 11:149–171, 2001.
5. Morten Winkler Jrgensen, Esben Hallundbk Ostergaard, and Henrik Hautop Lund. Modular atron: Modules for a self-reconfigurable robot. In *In proceedings of IEEE/RSJ International Conference on Robots and Systems, (IROS)*, pages 2068–2073. IEEE Press, Piscataway, New Jersey, USA, 2004.
6. Akiya Kamimura, Satoshi Murata, Eiichi Yoshida, Haruhisa Kurokawa, Kohji Tomita, and Shigeru Kokaji. Self-Reconfigurable Modular Robot–Experiments on Reconfiguration and Locomotion. In T. J. Tarn, editor, *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems IROS2001*, volume 1, pages 606–612. IEEE Press, Piscataway, New Jersey, USA, 2001.
7. A. Martinoli and F. Mondada. Collective and cooperative group behaviours: Biologically inspired experiments in robotics. In O. Khatib and J.K. Salisbury, editors, *Proc. of the Fourth Int. Symp. on Experimental Robotics ISER-95*, volume 223 of *Lecture Notes in Control and Information Sciences*, pages 3–10. Springer Verlag, Berlin, Germany, 1997.
8. Alcherio Martinoli. *Swarm Intelligence in Autonomous Collective Robotics: From Tools to the Analysis and Synthesis of Distributed Collective Strategies*. PhD thesis, Swiss Federal Institute of Technology in Lausanne (EPFL), Lausanne, Switzerland, 1999.
9. M. Mataric, M. Nilsson, and K. Simsarian. Cooperative multi-robot box-pushing. In *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 556–561. IEEE Computer Society, 1995.
10. F. Mondada, M. Bonani, S. Magnenat, A. Guignard, and D. Floreano. Physical connections and cooperation in swarm robotics. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS8)*, pages 53–60. IOS Press, 2004. Conference, Amsterdam, NL, March 10-14, 2004.
11. F. Mondada, G. C. Pettinaro, A. Guignard, I. Kwee, D. Floreano, J.-L. Deneubourg, S. Nolfi, L.M. Gambardella, and M. Dorigo. Swarm-bot: a new distributed robotic concept. *Autonomous Robots*, 17(2–3):193–221, 2004.
12. Lynne E. Parker. Alliance: an Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, 1998.
13. Mark Yim. *Locomotion with a Unit-Modular Reconfigurable Robot*. PhD thesis, Stanford university, Stanford, CA, 1994.

Timescale and Stability in Adaptive Behaviour

Christopher L. Buckley, Seth Bullock, and Netta Cohen

School of Computing, University of Leeds, UK
{christb, seth, netta}@comp.leeds.ac.uk

Abstract. Recently, in both the neuroscience and adaptive behaviour communities, there has been growing interest in the interplay of multiple timescales within neural systems. In particular, the phenomenon of neuromodulation has received a great deal of interest within neuroscience and a growing amount of attention within adaptive behaviour research. This interest has been driven by hypotheses and evidence that have linked neuromodulatory chemicals to a wide range of important adaptive processes such as regulation, reconfiguration, and plasticity. Here, we first demonstrate that manipulating timescales can qualitatively alter the dynamics of a simple system of coupled model neurons. We go on to explore this effect in larger systems within the framework employed by Gardner, Ashby and May in their seminal studies of stability in complex networks. On the basis of linear stability analysis, we conclude that, despite evidence that timescale is important for stability, the presence of multiple timescales within a single system has, in general, no appreciable effect on the May-Wigner stability/connectance relationship. Finally we address some of the shortcomings of linear stability analysis and conclude that more sophisticated analytical approaches are required in order to explore the impact of multiple timescales on the temporally extended dynamics of adaptive systems.

1 Introduction

Many of the model systems central to artificial life are explicit networks of simple interacting elements. Cellular automata (CA), artificial neural networks (ANNs) and random Boolean networks (RBNs), for instance, have become key tools in understanding what it is for a system to exhibit complex adaptive behaviour. Such models tend to be the subject of various different kinds of question. For example, the generation of different classes of dynamic behaviour (fixed, cyclic, complex, chaotic) has been of interest to CA and RBN researchers, e.g., [1,2] whereas those working with ANNs have been interested in questions of evolvability, problem solving and autonomous agent control, amongst others [3]. Interestingly, in answering these questions, the role of timescale within these systems has often been neglected. CA and RBNs typically comprise elements that share the same timescale (and updated with the same frequency),[4]. Similarly, while some continuous-time recurrent neural networks (CTRNNs) comprise neurons with explicit and varied timescales, this property has not received as much attention as others. For example, Beer [3] presents an extensive examination of

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 292–301, 2005.

the dynamics of recurrent CTRNN neurons, but only briefly mentions the impact of their time constants. This tendency to downplay timescale is somewhat surprising, since the natural adaptive systems that inspired these models typically involve processes and mechanisms that operate at multiple timescales. In particular, there is growing recognition that slow chemical processes within neural systems can be key to their ability to exhibit stable, sensitive, reconfigurable adaptive behaviour [5-7].

Here, we adopt an approach to understanding stability in complex networks inspired by classic cybernetics research, and adapt it to explore questions of timescale raised by this current work. First, a brief and selective account of the role of timescales in neural systems is presented, before a simple model exhibiting timescale-sensitive dynamics is detailed. Subsequently, a numerical approach to characterising the influence of timescale on stability is undertaken. The results are discussed and future directions are suggested.

1.1 Neuromodulation and Multiple Timescales

Neuromodulation is a term used diversely by neuroscientists to identify non-traditional processes acting alongside conventional neurotransmission. Although the term has been in use for over 20 years, the ubiquity of such processes has only recently been acknowledged. The action of a neuromodulator within the nervous system differs significantly from that idealised within the traditional connectionist paradigm: fast, point-to-point, excitatory/inhibitory [6]. Within neuroscience, there is a large and growing literature that associates slow, diffusive, modulatory, chemical mechanisms with a wide range of important adaptive capacities. Turrigiano [7], for instance, suggests that this type of mechanism is important for efficient lifetime adaptation within vertebrate nervous systems. Neuromodulators have also been implicated in triggering plasticity, regulating activity, governing reconfiguration, etc. [6]. However, conjectures on the role of neuromodulation in adaptation are not solely the province of the neuroscience community. There have also been treatments of this issue within the artificial life and adaptive behaviour communities [8,9]. For instance, the success of GasNets, a novel class of artificial neural network inspired by neuromodulation research [10], as an evolutionary robotics control architecture has generated a number of interesting theories regarding neuromodulation and adaptive behaviour [11]. GasNets consist of a traditional connectionist network over which the diffusion of neuromodulatory gases is modeled. The underlying network is embedded in a 2D space, where each neuron has the potential to emit gas, which diffuses over the network from a point source, affecting the properties of the gas-sensitive neurons that it comes into contact with. This gas mechanism is inspired by the neuromodulator nitric oxide (NO), which is small enough to pass freely through lipid tissue. The emission of NO is thought to be ubiquitous throughout the nervous system, but in general it is not accounted for in artificial models of neuronal systems.

Although GasNets have only been tested on a small range of tasks to date, the ease with which high-quality solutions can be evolved suggests that the pres-

ence of idealised neuromodulation may increase the evolvability of this class of control system across a range of real-world problems [12]. As yet there is little understanding of why this should be the case. While GasNets have been explored via a series of metrics, the contribution that neuromodulation makes to network evolvability remains unclear [13]. Aside from this postulated contribution to GasNet evolvability, the inclusion of idealised neuromodulatory mechanisms in a control system could result in greatly enhanced adaptive properties. However, it is unclear whether these benefits are due to the specifics of the chosen abstraction or more fundamental principles underlying neuromodulation. There is some evidence that it is the combination of fast (neurotransmission) and slow (neuromodulation) processes that may be responsible [14]. Indeed, the slow nature of neuromodulation appears crucial to many of its postulated roles. Whether regulating the gross activity in a neural circuit, or maintaining a neural variable within critical bounds via homeostatic plasticity [7], or switching between different modes of circuit behaviour dynamics (e.g., the switch between swimming and the escape reflex in *Tritonia*, [15]), neuromodulators are often best considered as slow processes that parameterise a fast sub-system. Understanding how to model this interaction across temporal hierarchies remains an open question.

Of course, the presence of explicitly slow elements or processes is not necessary in order to allow a system to exhibit multiple timescales. The flow of activation through a large recurrent network of fast elements may allow different timescales to arise. For instance, Harvey and Thompson [16] evolved circuitry to discriminate between slow oscillatory inputs where the intrinsic timescale of the components (a few nanoseconds) is five orders of magnitude shorter than the dynamics exhibited by the evolved circuit. Furthermore, in small systems, saddle node or homoclinic bifurcations can give rise to slow dynamics even if the underlying nodes are intrinsically fast [17]. For example, in most models of spiking neurons the explicit timescales are fast, usually on the order of $10ms$ or less [6], yet in many cases the dynamics of interest extend well beyond these characteristic timescales. However, given that neural substrates support adaptive behaviour at many different temporal scales and that neuromodulators act on a range of timescales typically slower than that of neurotransmission, it seems intuitive that there may be some value in this explicit combination of multiple timescales.

2 Stability Criteria for Complex Networks

In a now classic study, Gardner and Ashby [18] investigated stability criteria for large complex systems in terms of the effect of connectivity on the tendency of a system to exhibit a stable point attractor. The relationship between a network's structure and its stability has been of long standing importance, particularly in the field of ecology [19]. At the time, biologists typically assumed that the stability of an ecosystem would increase with its biodiversity (due to mean field averaging). The same issue has significance for systems ranging from traffic networks to the human brain. In each case, Gardner and Ashby argued, we should

not necessarily expect to observe stability as systems grow in size. Their numerical results characterised the way in which networks of interacting elements become less stable as their interconnectivity increased. This tendency towards stability was subsequently formalised by May [20], who derived a threshold for stability in terms of the mean-square of the strength of the connections and the degree of interconnectivity. In both these studies, the systems are assumed to comprise elements that share a single intrinsic timescale.

Gardner and Ashby [18] and May [20] considered the stability of a linear system $\mathbf{y} = (y_i, i = 1...N)$, given by

$$\dot{y_i} = -y_i + \sum_{j=1}^{N} \omega_{ij} y_j \qquad \text{in vector form :} \qquad \frac{d\mathbf{y}}{dt} = \mathbf{A}\mathbf{y} \tag{1}$$

Here, $\mathbf{A} = \mathbf{\Omega} - \mathbf{I}$, where $\mathbf{\Omega} = (\omega_{ij})$ is a matrix of weighted interaction strengths and $\mathbf{I}$ is simply the identity matrix. Such a system is said to be stable when every eigenvalue of $\mathbf{A}$ has a negative real part [21]. Gardner and Ashby [18] employed a numerical method to discover the stability of an ensemble of random networks, varying network size, N, and network connectivity, C (the probability that any entry of the weight matrix $\mathbf{\Omega}$ is non-zero or, equivalently, the probability that any two elements interact). They were able to demonstrate that stability could be compromised by high connectivity.

To derive a threshold for stability, May [20] used analytical results from the field of random matrix theory [21,22]. He drew the entries of $\mathbf{\Omega}$ from a statistical distribution with zero mean and a mean-square value, α. He then derived a critical threshold above which any network has a high probability of instability. Explicitly, he stated that in the limit of large system size ($N \gg 1$), a system is almost certainly unstable if $NC\alpha^2 > 1$.

This result, generally referred to as the May-Wigner stability theorem, corresponds well with Gardner and Ashby's original findings and still holds as a very important threshold [23]. It has been extended recently to demonstrate that the result stills holds for systems in which connections between elements exhibit time delays [24]. However, as yet, the influence of timescale, as distinct from time delay, has not been explored. Recent work within neuroscience and adaptive behaviour suggests that systems involving processes on multiple timescales readily exhibit important classes of adaptive behaviour. Here we apply the approach introduced by Gardner and Ashby [18] and formalised by May [20] to such systems.

3 Timescale in a Two-Node System

The analysis described above assumes linearity, yet it is possible to apply the results to non-linear systems if we restrict our attention to behaviour in the vicinity of a specific equilibrium. In this case, we can consider the local behaviour around this equilibrium and determine the stability of the system under a (vanishingly small) perturbation. This process is known as linear stability analysis. It will tell

us about the local asymptotic behaviour of a non-linear system around a particular equilibrium but tell us nothing about global stability. For example, while a limit cycle cannot be said to be locally stable, it may be globally stable such that under perturbation the system always settles to the same cyclic trajectory.

May's result has been criticised because it relies on this linearization around equilibrium, which is thought to make it inapplicable where perturbations are large or systems exhibit limit sets of higher dimension than a fixed point. While this issue remains open, recent calculations of global dynamics have obtained the May-Wigner stability thresholds as thresholds for global system stability. These results suggest that the May-Wigner theorem may be more universal than originally expected[23]. So, while this technique has restricted application to non-linear systems, it may still has the potential to deliver general insight into the dynamics of complex systems.

We will consider a system of equations used to describe continuous-time recurrent neural networks (CTRNNs). The CTRNN is commonplace throughout neuroscience (as a leaky integrator) and evolutionary robotics [3].

$$\dot{y_i} = -\frac{y_i}{\tau_i} + \frac{\tanh\left[\sum_j \omega_{ij} y_j + \theta_i\right]}{\tau_i} \tag{2}$$

Here y_i represents activation at the i^{th} neuron; ω_{ij} is a weight on the connection between neurons i and j; θ_i is the bias value at the i^{th} neuron; and τ_i is the time constant of the i^{th} neuron, which defines the rate of leakage or decay of activation. The equation is forward integrated with a simple Euler step method with time slices of $dt = 0.005$. Note that τ represents the explicit timescale of each of the units and it is this parameter that we will concern ourselves with in this work. In this formulation, the sigmoidal transfer function is a hyperbolic tangent rather than the more familiar exponential sigmoid (see e.g., REF [3]. Note that, here, activation does not represent the membrane potential of a neuron, but rather the firing *rate*, or mean number of spiking events per unit time, averaged over some appropriate time window. In general we can think of the CTRNN equation as a re-description of the firing rate of a given neuron (or ensemble) averaged over some window, τ.

We will first consider a simple two-node system described by equation (2). To determine the linear stability of this system, we must first calculate the coordinates of its equilibrium point. This is located at the intersection of the system's nullclines, each defined by $\dot{y_i} = 0$. Second, we must calculate the Jacobian of the system at equilibrium,J, given by equation (3), (further details can be found in Refs. [17] and [3].)

$$J = \begin{pmatrix} \frac{d\dot{y_1}}{dy_1} & \frac{d\dot{y_1}}{dy_2} \\ \frac{d\dot{y_2}}{dy_1} & \frac{d\dot{y_2}}{dy_2} \end{pmatrix}_{\bar{y_1}, \bar{y_2}} \tag{3}$$

Here, $\bar{y_1}$ and $\bar{y_2}$ are the equilibrium activation values, and the matrix therefore represents the instantaneous interaction between each element around the equilibrium point, and can be analytically calculated. Under these conditions, this

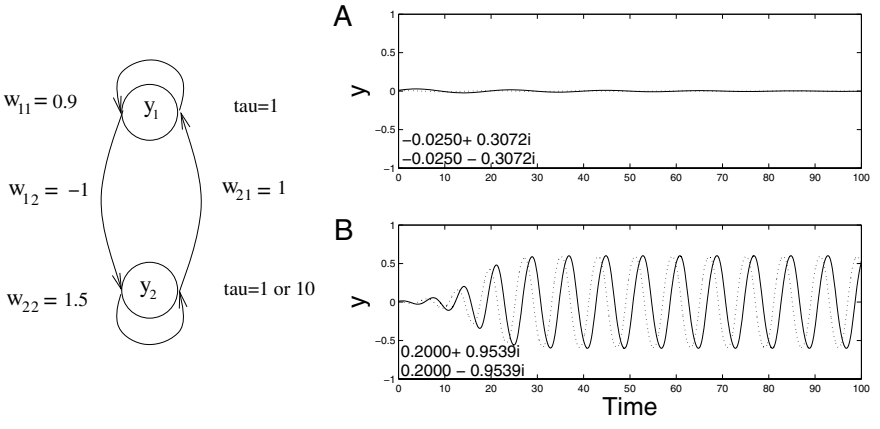


Fig. 1. Variation in the behaviour of a simple two-node circuit with recurrent links (parameterized as shown, left), due to manipulating the timescale of its component elements. In each case, the system is released from an initial condition ($y_1 = y_2 = 0.01$) in the vicinity of the equilibrium at $\bar{y_1} = \bar{y_2} = 0$. A. $\tau_1 = 1, \tau_2 = 10$: The system exhibits stability. B. $\tau_1 = \tau_2 = 1$: the system diverges from equilibrium to a limit cycle. Eigenvalues of the Jacobian for each system are shown alongside the plots.

matrix is equivalent to **A** in May's formulation. We can now determine whether the system is stable by requiring that the real parts of each eigenvalue of the matrix are negative.

In Refs. [18] and [20] the timescales of all the elements within a system are assumed to be equal. Here we consider the consequences of relaxing this assumption. In general determining the contents of the Jacobian matrix requires us to calculate complex terms that depend on the first order differential of the CTRNN sigmoidal transfer functions. By stipulating that $\theta_1 = \theta_2 = 0$ we guarantee that there is a system equilibrium at $\bar{y_1} = \bar{y_2} = 0$, which simplifies the Jacobian, thus:

$$J = \begin{pmatrix} \frac{\omega_{11} - 1}{\tau_1} & \frac{\omega_{12}}{\tau_1} \\ \frac{\omega_{21}}{\tau_2} & \frac{\omega_{22} - 1}{\tau_2} \end{pmatrix} \tag{4}$$

We can rewrite equation (3) in vector form equivalent to equation (1) for a system with multiple timescales as $\mathbf{A} = (\mathbf{\Omega} - \mathbf{I})\tau^{-1}$, where τ is a vector of the damping times, τ_i, for each of element. The question here is what effect this has on the dynamics? To understand this we will consider an example of a coupled two-node system parameterized as illustrated in figure 1.

Figure 1 depicts the behaviour of the coupled system for $\tau_2 = 1$ and $\tau_2 = 10$ (holding $\tau_1 = 1$ constant) from the same initial conditions ($y_1 = 0.01, y_2 = 0.01$). For $\tau_2 = 10$ the system is locally stable, converging to equilibrium after a small perturbation. In contrast, for $\tau_2 = 1$ the equilibrium at $y_1 = y_2 = 0$ is unstable. Even though the system is initially perturbed only a small distance from this

equilibrium, the trajectory diverges to a limit cycle. In fact, as we alter τ_2 the system undergoes a subcritical Hopf bifurcation [17]. Is this bifurcation reflected in the linear stability analysis? From equation (4) we can determine that the real parts of each eigenvalue change from positive to negative as we increase τ_2 (see figure 1), indicating a transition from local instability to local stability.[1]

In this simple case, timescale (as well as connectivity and weight strengths) affects system stability. It is interesting to note the direction of this influence—increasing timescale separation increases system stability. This begs the question: what effect does timescale have on larger systems, and does it interfere with the relationship described by Gardner and Ashby, and formalised by May?

4 Larger Systems

In the previous section, we have outlined how linear stability analysis can shed light on the dynamics around an equilibrium position in a non-linear system. For the small system considered above, varying the timescale parameters, τ, brought about a Hopf bifurcation, altering the system's dynamics such that it ceased to exhibit a stable equilibrium. Could timescale have a similar effect on the stability of larger systems? Gardner and Ashby [18] and May [20] considered the effect of both connectivity, C, and mean-square weight value, α, on stability, but assumed that the damping time of each of the system's elements was unity. In this section we will relax this assumption. To achieve this, we will establish numerically the relationships between probability of stability and both C and α for networks with all $\tau_i = 1$, and compare this with the same relationships for networks with τ_i uniformly distributed over three orders of magnitude.

The basic form of these relationships, depicted in figure 2, is intuitive. At low α or C, networks have a high probability of stability, which decreases as α or C increase. Figure 2's vertical dotted lines represent the critical threshold derived by May. Predictably, the correspondence between the (asymptotically derived) threshold and the numerical results increases with network size, as does the steepness of the numerically derived "phase transition". However, less predictably, there appears to be little difference between the stability of networks comprising elements with shared, unitary timescale and networks comprising elements with widely varying timescale. In contrast to the example given in section 3 above, multiple timescales have little effect on the stability threshold, or on the general character of the relationship.

Our paired design allows us to confirm that if a network below the May-Wigner threshold is stable with unitary timescale elements, the same network will generally be stable if those timescales vary widely. However, for networks above the May-Wigner threshold, in all plots the probability of stability in timescale-separated networks is slightly, but systematically, *lower* than the probability of

[1] As this analysis only concerns the local behaviour around the equilibrium, it tells us nothing about subsequent trajectories. Nevertheless, the bounded nature of this system and the fact that it can only exhibit one point equilibrium guarantees that, where the original equilibrium is unstable, a cyclic attractor will surround it.

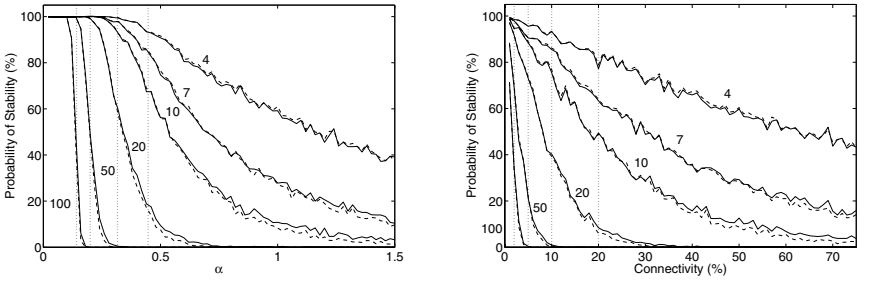


Fig. 2. Probability of stability versus. (a) the root mean square of network weights, α, and (b) network connectivity, C, for networks of size 4, 7, 10, 20, 50 and 100 nodes (1000 random networks per data point). For (a), $C = 50\%$. For (b), $\alpha = 1$. Solid curves depict results for networks with unitary τ values, dashed curves for the same networks with τ values uniformly distributed across three orders of magnitude. Vertical lines denote the stability threshold as predicted by the May-Wigner theorem for networks of 100, 50, 20 and 10 nodes (reading left to right).

stability in equivalent unitary networks. This may indicate that the presence of multiple timescales *encourages* the transition to instability. This effect is small, less than 1% for all network sizes. Although this difference seems negligible in the context of the overall character of the relationship, it would be interesting to investigate its root cause since it is in opposition to the effect of timescale separation demonstrated in section 3.

So far, we have concerned ourselves only with the real parts of a network's eigenvalues, since these reveal the presence of local stability. While the introduction of multiple timescales has little effect on the probability that these real parts are all negative (indicating local stability), it does have an effect on the imaginary parts of these eigenvalues, which are far more likely to be non-zero in this case. In a simple coupled system, these imaginary parts indicate the manner in which the system transitions to or from equilibrium. If the imaginary parts are zero, the equilibrium is said to be a *node*, otherwise it is a *spiral* [17].

The increase in the number of non-zero imaginary eigenvalue parts brought about by the introduction of multiple timescales implies that trajectories around the equilibrium have little or no curvature. We can understand this in terms of the strength of the effects of the different elements that comprise a network. Because each element's entry in the Jacobian matrix (3) is scaled by its inverse timescale, i.e., by $\frac{1}{\tau_i}$, slower elements will have a weaker instantaneous influence. Weakening or strengthening an element's influence will not tend to affect local stability, since even a weak effect can displace a system from equilibrium. However, the short-term behaviour of the system will appear to be dominated by fast elements, although slow elements may have a large effect in the long term.

This observation is reminiscent of Ashby's (1960) *temporary independence*, [25], used to describe how trajectories in the phase space of a complex system may evolve over low-dimensional manifolds if certain variables remain practically

constant over some period of time. The utility, in this context, of a distinction between interdependence over the short- and long-term is also reminiscent of Simon's [26] attempt to define functional modularity.

5 Conclusion

We have demonstrated that, in at least one example, altering the explicit timescale of a network component can effect a transition between stability and instability, despite connectivity and weight parameters remaining fixed. Conversely, we have shown that Gardner and Ashby's stability/connectance relationship and May's critical threshold are largely unaffected by the presence of multiple timescales.

In order to characterise the influence of timescale more satisfactorily, we must move beyond this initial linear stability analysis, and develop tools that allow us to explore the temporally extended non-equilibrium dynamics of systems exhibiting multiple timescales. One potential avenue is the extension of statistical, information-theoretic measures of interdependence, such as mutual information [27], to the task of determining whether sub-systems that are temporally separated might be functionally modular in the sense of Simon [26] or Watson [28]. Such modularity is hinted at by some of the results presented here, and would go a long way toward accounting for the different ways in which neuromodulation has been implicated in underpinning temporally extended adaptive behaviour.

References

1. Kauffman, S.: The Origins of Order. Oxford University Press, Oxford (1993)
2. Gershenson, C.: Classification of random Boolean networks. In Standish, R.K., Bedau, M.A., Abbass, H.A., eds.: Artificial Life VIII: Proceedings of the Eight International Conference on Artificial Life, MIT Press (2002) 1–8
3. Beer, R.D.: On the dynamics of small continuous-time recurrent neural networks. Adaptive Behavior 3 (1995) 471–511
4. Di Paolo, E.A.: Searching for rhythms in asynchronous Boolean networks. In Bedau, M.A., McCaskill, J.S., Packard, N.H., Rasmussen, S., eds.: Seventh International Conference on Artificial Life, MIT Press, Cambridge, MA (2000)
5. Poggio, T.A., Glaser, D.A., eds.: Exploring Brain Functions: Models in Neuroscience. John Wiley and Sons, New York (1993)
6. Katz, P.S., ed.: Beyond Neurotransmission: Neuromodulation and its Importance for Information Processing. Oxford University Press, Oxford (1999)
7. Turrigiano, G.G.: Homeostatic plasticity in neuronal networks: The more things change, the more they stay the same. Trends in Neuroscience 22 (1999) 221–227
8. Doya, K.: Metalearning and neuromodulation. Neural Networks 15 (2002) 495–506
9. Williams, H.: Homeostatic plasticity in recurrent neural networks. In Schaal, S., Ijspeert, A., Billard, A., Vijayakumar, S., Hallam, J., Meyer, J.A., eds.: Eighth International Conference on the Simulation of Adaptive Behavior, MIT Press, Cambridge, MA (2004) 344–353

10. . Husbands, P., Philippides, A., Smith, T.M.C., O'Shea, M.: The shifting network: Volume signalling in real and robot nervous systems. In Kelemen, J., Sosik, P., eds.: Sixth European Conference on Artificial Life, Springer, Heidelberg (2001) 23–36
11. Philippides, A.O., Husbands, P., Smith, T.M.C., O'Shea, M.: Fast and loose: Biologically inspired couplings. In Standish, R.K., Bedau, M.A., Abbass, H.A., eds.: Eighth International Conference on Artificial Life, MIT Press, Cambridge, MA (2002) 292–301
12. Smith, T.M.C., Husbands, P., Philippides, A.O., O'Shea, M.: Neuronal plasticity and temporal adaptivity: GasNet robot control networks. Adaptive Behavior 10 (2002) 161–184
13. Smith, T.M.C., Husbands, P., O'Shea, M.: Not measuring evolvability: Initial exploration of an evolutionary robotics search space. In: Congress on Evolutionary Computation. IEEE Press (2001) 9–16
14. Buckley, C., Bullock, S., Cohen, N.: Toward a dynamical systems analysis of neuromodulation. In Schaal, S., Ijspeert, A.J., Vijayakumar, A.B.S., Hallam, J., Meyer, J.A., eds.: Eighth International Conference on Simulation of Adaptive Behavior, MIT Press, Cambridge, MA (2004) 334–343
15. Hooper, S.L.: Neural circuits: Functional reconfiguration. In: Nature Encyclopedia of Life Science, Nature Publishing Group, London (2001)
16. Harvey, I., Thompson, A.: Through the labyrinth evolution finds a way: A silicon ridge. In Higuchi, T., Iwata, M., Weixin, L., eds.: First International Conference on Evolvable Systems, Springer-Verlag, Heidelberg (1997) 406–422
17. Strogatz, S.H.: Nonlinear Dynamics & Chaos. Addison-Wesley, Reading MA (1994)
18. Gardner, M.R., Ashby, W.R.: Connectance of large dynamic (cybernetic) systems: Critical values for stability. Nature 228 (1970) 784–784
19. McCann, K.S.: The diversity-stability debate. Nature 405 (2000) 228–233
20. May, R.M.: Will a large complex system be stable. Nature 238 (1972) 413–414
21. Mehta, M.L.: Random Matrices. Academic Press, New York (1967)
22. Wigner, E.P.: Gruppentheorie und Ihre Anwendung auf die Quantenmechanik der Atomspektren, trans. J. J. Griffin. Academic Press, New York (1959)
23. Sinha, S., Sinha, S.: Evidence of universality for the May-Wigner stability theorem for random networks with local dynamics. Phy. Rev. Let. E 71 (2005) 1–4
24. Jirsa, V.K., Ding, M.: Will a large complex system with time delays be stable. Physical Review Letters 93 (2004) 070602
25. Ashby, W.R.: Design for a Brain. Chapman and Hall, London (1960)
26. Simon, H.A.: The Sciences of the Artificial. MIT Press, Cambridge, MA (1969)
27. Tononi, G., Edelman, G., Sporns, O.: Complexity and coherency: integrating information in the brain. Trends in Cognitive Sciences 2 (1998) 474–483
28. Watson, R.A.: Modular interdependency in complex dynamical systems. In Bilotta, E., ed.: Workshop Proceedings of Alife VIII, MIT Press, Cambridge, MA (2003)

Whisker-Based Texture Discrimination on a Mobile Robot

Miriam Fend

Artificial Intelligence Laboratory,
University of Zurich, Andreasstrasse 15, 8050 Zurich, Switzerland
fend@ifi.unizh.ch
http://www.ifi.unizh.ch/~fend

Abstract. Sensing in the dark is a useful but challenging task both for biological agents and robots. Rats and mice use whiskers for the active exploration of their environment. We have built a robot equipped with two active whisker arrays and tested whether they can provide reliable texture information. While it is relatively easy to classify data recorded at a specified distance and angle to the object, it is more challenging to achieve texture discrimination on a mobile robot. We used a standard neural network classifier to show that it is in principle possible to discriminate textures using whisker sensors even under real-world conditions.

1 Introduction

When light is dim or fading, tactile information becomes more and more important. In nature, many night-active animals such as rodents, cats or oppossums have developed an exquisite tactile organ, the whiskers. With their large mystacial whiskers, rats for example not only navigate to avoid obstacles, but they are also able to discriminate different textures and shapes [4]. Behavioral studies in rats have shown that their ability to discriminate surface structures with the whiskers is comparable to ours using our fingertips [11] [5]. Unravelling the information coding in the rat whisker system has recently attracted different researchers both from biology [18] [17] [3] [2] and from the field of robotics [10] [20] [22] [19] [15]. Theoretical studies have analyzed the properties of whisker vibrations [16] [9] [14] and their implications on neural coding and learning of simulated receptive fields [13] [12].

So far, tactile stimuli have largely been acquired by keeping parameters such as distance and orientation of the whiskers constant with respect to the texture (as in [20] [9]). Although it is reasonable to assume that animals can position their head appropriately, they are also able to discriminate textures from far away when forced to do so. One of the main differences between analyzing recorded data and using a behaving robot is that different parameters such as distance and angle towards the texture are not necessarily well defined. Thus, it is important to record data with different parameters and identify features significant for the discrimination of textures. Such features are necessary for the construction of a behaving system capable of showing discriminatory behavior comparable to a trained rat.

To our knowledge, so far only one study has conducted experiments on texture discrimination with a mobile robot [20]. In their experiment, the robot showed a wall

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 302–311, 2005.

following behavior stimulating the whisker sensors by moving them across the wall. When a texture was encountered, the robot learned to avoid the wall based on the activity pattern of its neural network. Following the wall not only generates input, it also controls for the distance and angle at which a tactile pattern is sensed. The input to the neural system is thus more reliable and reproducible than at random orientations.

In the series of experiments presented in this paper, we want to consider a more general case, namely whether classification is possible even if a texture is explored from different angles and distances. Furthermore, the robot generates sensory stimulation not only by moving the whole body, but also by moving the whiskers actively. We have approached this question twofold: first, we recorded different textures from different angles and distances and trained a neural network to classify these textures. In a second series of experiments, we let a robot explore an environment equipped with different textures and trained a network with these self-acquired data. During a separate testing phase, the classification of the sensory input was recorded and evaluated.

2 Materials and Methods

The goal of this series of experiments was to assess the robustness and the discriminatory power of the whisker sensors under real-world circumstances. Detailed data analysis has been performed elsewhere [9]. We used a microphone-based whisker sensor with natural rat whiskers as described in [16]. A single whisker hair of approximately 5 cm is glued to a capacitor microphone. Mechanical stimulation is thus transduced to a deformation of the microphone membrane. The resulting signal is amplified and recorded by the computer. Six such whiskers are assembled in an array of two rows with three whiskers. They can be moved actively by one servo motor to perform a periodic synchronous sweep at a frequency of 1 Hz. The construction of the whisker array has been described in detail in [8].

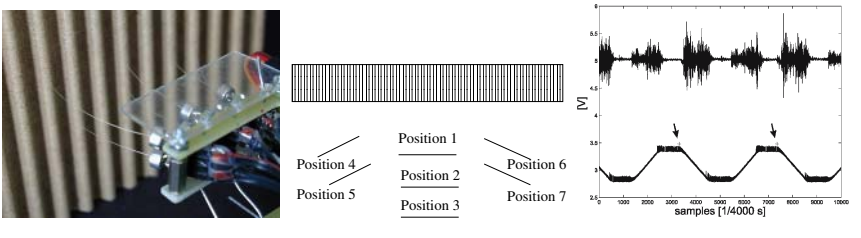


Fig. 1. a) Photograph of the data collection setup with rough carton. The 6 whiskers of the artificial whisker array can be moved synchronously by one servo motor. The whiskerarray was placed at different distances and angles towards the texture. **b)** Schematic of the layout of the seven positions at which data was recorded with respect to the texture (indicated as a striped bar) **c)** Example of one sweep of raw data and the recorded motor signal. The borders between sweeps as extracted by the algorithm are marked with arrows.

2.1 Data Acquisition

We collected a dataset containing four different textures: 1) smooth metal, 2) sandpaper 400, 3) sandpaper 80 and 4) rough carton recorded at seven different positions (see figure 1(b)). At position 1, the base of the whisker sensor is at a distance of 2 cm from the texture. The positions in one column are each 1 cm apart. The whiskers were actively moved across the surface of the texture and the position of the servo motor was recorded simultaneously. Data acquisition was performed using a National Instruments Data Acquisition Card (DaqCard 6036E) at 4 kHz per channel.

For the robot experiments we used an open environment. Half of the surface was lined with a rough carton surface, the other half was left blank, displaying a smooth metallic surface.

2.2 Feature Extraction and Discrimination Capabilities of Recorded Data

Previously, we have shown that it is possible to generate texture specific signatures from power spectra of whisker signals (see [9]). Such a signature relied on several sweeps and covered frequencies up to 1 kHz. For a system behaving in real time, we sought to reduce the dimensionality of the input vector further. Three different preprocessing methods for feature extraction were tested: Spectrotemporal analysis, fourier transform convolved with a Blackman window of 70 data points (57 Hz) and raw data also convolved with a window of 57 Hz. In all three cases, the dimensionality was reduced to

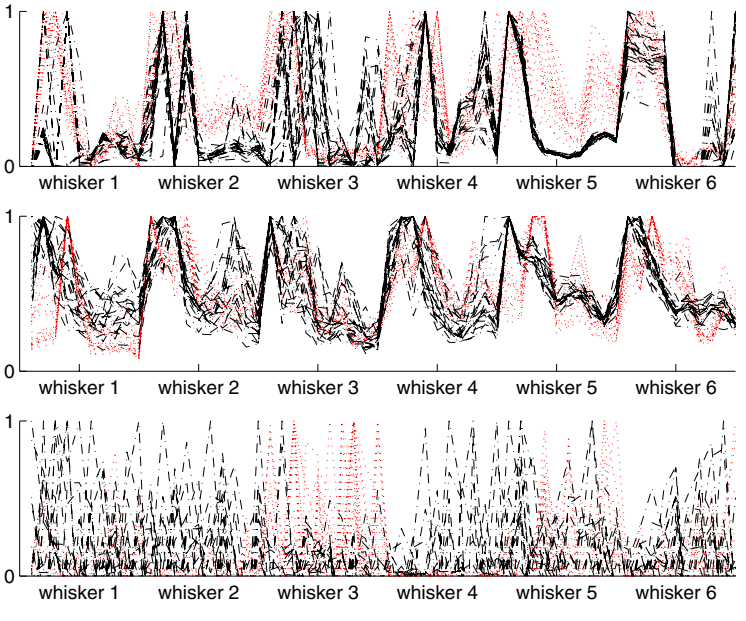


Fig. 2. Cumulated feature vectors of twenty sweeps in one position of texture 1 and texture 4. The dotted line indicates texture 1, dashed line texture 4. The preprocessing used was **Top row:** Smoothed raw data, **middle row:** fft and **bottom row:** PCA components after a spectrotemporal analysis.

10 values per whisker yielding a feature vector with 60 values. The raw data and the fourier transformed data were divided in 10 windows (the first 750 ms of each sweep and the frequencies between 1 and 1000 Hz) of 75 ms and 100 Hz respectively. Then the highest value of this window was passed as input to the network. Examples of 20 such input vectors of two different textures are shown in figure 2.

2.3 Training the Neural Network

To identify and evaluate different features, a standard backpropagation network was used to classify previously recorded textures. Please note that the purpose of this experiment was not to postulate a specific biologically inspired architecture, but to evaluate the potential of the features used and the setup as a whole under real-world conditions. Any other statistical classification algorithm could have been used as well. Training was done using the Levenberg-Marquardt algorithm as implemented by the Matlab Neural Network Toolbox [1]. For all neural networks described in this paper, we trained ten runs with different random initializations and between 10 and 14 hidden layer neurons.

Since the whiskers were stimulated by actively sweeping over the surface, the proprioceptive signal from the motor identified the repeating elements. Multiple sweeps of the same texture were thus extracted from one continuous stream of input. One such sweep together with the motor signal is shown in figure 1(c). Together with the remaining five whiskers, this constitutes one sample of input for feature extraction and subsequent neural network training.

For each texture, one minute of data was recorded at seven different positions systematically varying the distance and angle of the whisker array with respect to the presented texture (figure 1(b)). A second set of data was recorded separately to be used for testing the network.

2.4 Evaluation of Network Performance

To test the classification and generalization, each trained network was simulated with the test data and a hit matrix (as in figure 4(a)) was computed by determining the output neuron responding most strongly and comparing it to the desired output neuron. From the hit matrix, the percentage of correctly classified samples was computed. After feature extraction using fourier transformation with subsequent dimensionality reduction

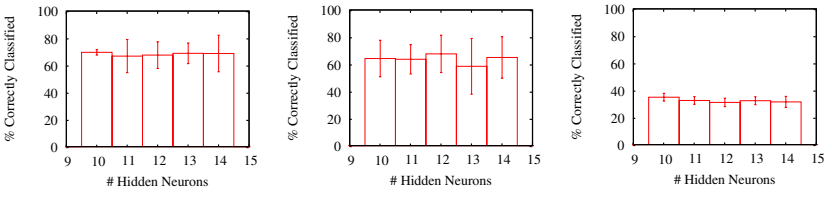
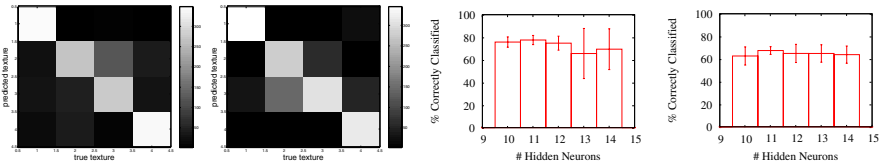


Fig. 3. Mean percentage of correctly classified samples using **a)** smoothed raw data and the ten highest values of each whiskers in blocks of 75 ms. **b)** FFT preprocessing **c)** spectrogram preprocessing with subsequent PCA.



(a) (b) (c) (d)

Fig. 4. Left Sample hit matrix on all recorded positions and textures with **a)** FFT preprocessing and **b)** smoothed raw data. The textures are from 1 to 4: smooth metal, sandpaper 400, sandpaper 80 and cardboard. **Right** Mean percentage of correctly classified test samples recorded with a mobile robot. **c)** Left whisker array and **d)** right whisker array. The textures to be discriminated were smooth metal vs. rough carton.

as well as the temporal analysis of the raw data, the neural network was able to classify not only the training set but also the test set (figure 3(a) and 3(b)). The best classification for raw data was 75 %, for spectral analysis (fft) it was 74 %. Usually, about one of the random initializations resulted in a network unable to classify the testdata above chance. This is the reason for the rather large errorbars in figure 3(b).

Figure 3(a), 3(b) and 3(c) show the mean number of correct responses for the three different types of feature extraction for 10 different random seeds and different numbers of hidden neurons. Spectrotemporal analysis followed by principal component analysis was not able to learn to discriminate the four textures, mean correct responses range between 25 % and 39 %.

Figures 4(a) and 4(b) show the hit matrices for the testdata with a sample neural network. Bright color indicates many entries. The bright diagonal shows that the network classified the textures correctly in most cases. More interesting is the interpretation of misclassifications: most mistakes occured for the two sandpapers (textures 2 and 3). Smooth metal and rough carton were rarely confused. The distinction between these two textures was especially clear between feature extraction using spectral analysis, therefore it was used in the robot experiments.

3 Classification of Data Recorded on a Mobile Robot

First tests with the robot were conducted using the same features and neural network structure as determined to be appropriate with recorded data. However, when the robot did not use any sensory feedback to adjust its position with respect to the encountered surface, often it did not get stimulation in more than two whiskers. Data recorded under such conditions did not result in successful classification (data not shown). Therefore, the whisker data was used to roughly position the robot such that at least four whiskers were stimulated.

For this behavior, the robot was equipped with a few motor primitives: It explored the environment while whisking actively for obstacles and explorable textures. Upon contact, the robot stopped and acquired a few whisks of data. Depending on this sensory input, it either logged data or repositioned slightly with a fixed turning behavior in order to achieve stimulation in at least four whiskers before acquiring data. The training signal

necessary for the backpropagation algorithm was delivered manually. In unfortunate spots such as ambiguous corners, the robot was repositioned manually.

After a total of 150 encounters which were shared about equally between the two whisker arrays, the first 3/4 of the encounters of each side of the robot were used to train the neural network, the last fourth of encounters was used to test the performance. Please note that between every encounter, the robot moved for a minimum of 2.5 s including turning on the spot. This ensured that each instance of acquiring data was actually done at a new orienation and at a different spot. On average, the left whisker array classified correctly more often than the right whisker array. The mean values on the left side ranged between 65 to 76 % correctly classified samples with the best network classifying 85 % of the test samples correctly (figure 4). The right whisker array on average classified between 63 % and 67 % percent of the samples correctly. The maximum of correctly classified samples was 76 % (figure 4). The differences found between the two whisker arrays can depend on several factors which cannot be decided on the basis of the current experiments. Possibly, the quality of the whisker sensors varies. Another source of variation is that the robot acquires data on its own and thus it may be that one side accidentally records data more apt for classification.

3.1 Behavioral Experiments with the Robot

The same neural network structure was used for the robot as was tested previously in the simulation described above. For each of the two whisker arrays with six whiskers a neural structure was created: this right and left hemisphere were fed with the signals from the respective whisker arrays and trained individually. During a behavioral testing phase, the previously trained robot explored the environment. Upon contact with a texture, it was palpated for 9 seconds of which five sweeps were used for classification. Depending on the resulting classification, the robot responded by turning by 30° or by 120° away from the texture. Given this behavior, we expect the robot to spend more time in that half of the arena, where the turning angle is smaller. The resulting trajectory should thus cover the respective part of the arena more closesly. To evaluate the robot performance, each run was recorded with an overhead camera and the robot was automatically tracked using the KLT library [21]. As a control condition, the robot behaved as described above, but instead of using sensory input for classification, the type of texture was supplied by the experimentor. Here, only slippage of the wheels or physical hindrance e.g. due to the cables can possibly induce deviation from a perfect behavior.

In the actual experiment, the robot classified whisker input with the neural networks trained as described above. To ensure that a behavioral pattern was actually induced by correct classifications and was not an artefact of the allocation of texture type and turning angle, this allocation was also switched.

3.2 Results of the Behavioral Experiments

In the control condition shown in figure 5(a) it is apparent that the robot spends much more time close to the smooth metal. It also reliably turns away from the carton. This is due to the different preprogrammed turning angles. Reversing the angles also reverses

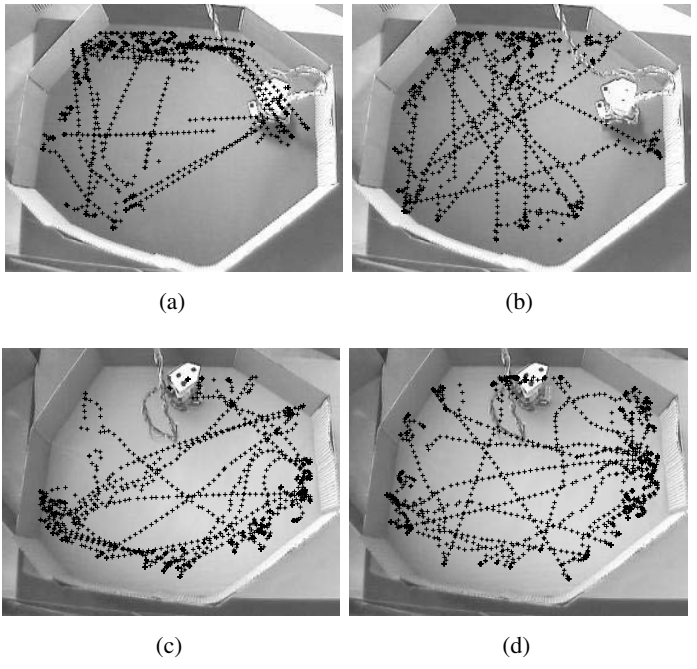


Fig. 5. Trajectories of a single run. Each cross indicates the robot position during one frame of a consecutive image sequence. The background shows the actual robot arena with the robot as seen from an overhead camera. The bottom and the right wall are coated with rough carton, the upper and left walls are made of smooth metal. **Top row:** The robot turns from rough carton at a larger angle than from the texture classified as smooth metal. **a)** Classification supplied by the experimentor and **b)** classification according to sensory input. **Bottom row** The robot turns stronger from smooth metal (120° angle) than from rough carton. **c)** classification supplied by the experimentor and **d)** classification according to sensory input.

the overall impression (figure 5(c)). During the actual experiment, the classification depended solely on the sensory input acquired by actively whisking any surface encountered during exploration. Figure 5(b) shows such a run: the robot spends more time close to the metal coated walls. This is due to the lower angle with which it turns from the texture classified as metal. Larger turning angles can be seen well for encounters with the rough carton coated walls.

The same holds true when the turning angles are reversed (figures 5(c) and 5(d)). Here, the robot turns with a 30° angle from rough carton and with a 120° angle when palpating smooth metal.

4 Discussion and Future Work

Tactile discrimination based on whiskers is still a young research area. The experiments described above try to fathom the potential of artificial whiskers for haptic sensing both statically and on a robot. For this purpose, a standard classifier was used, namely a backpropagation network.

Since whiskers are potentially very interesting tactile sensors for robots, the main focus of the experiments was to assess how reliable whisker-based classification is without strict control of position and orientation. The results of neural network simulation of data recorded at different but defined positions are promising. Even with only few inputs and a standard preprocessing such as fourier transformation, classification of four different textures with about 70 % correctly recognized textures based on only one sweep has been achieved.

To test whether this would hold true for the continuous space of possible distances and orientations on a mobile robot, robotic experiments were conducted. In this series of experiments it became apparent that it is more difficult to achieve classification behavior under real world conditions. Firstly, sensory feedback based on whisker input had to be introduced to avoid active exploration in situations when only one or two whiskers touched the surface. Having limited the range of possible positions to those, where at least four of six whiskers were activated, test data could be classified to some extend, but not without mistakes.

Based on these results, a lot of experiments can be proposed. For example, we want to test the whisker-based texture discrimination of the robot in a behavioral task comparable to experiments on rats. We have already built a maze with variable number of arms. The robot should be able to chose specific arms based on textural information at the walls of each arm. For this task it will probably be necessary to improve the reliability and the discriminatory capability of the system. While we cannot exclude that the preprocessing chosen for these experiments is not optimal, we believe that to achieve more reliable classification sensory-motor coordination might be used on two levels. Firstly, feedback from the whiskers could be used adaptively to orient the body of the robot appropriately with respect to the texture. Rats for example are reported to prefer a distance of 2 cm from their whiskers to an object or texture [7]. Secondly, the whisking behavior itself could be influenced by sensory feedback. Varying the speed or amplitude of whisking could possibly help to resolve ambiguities. Again, there is evidence from behavioral rat studies that the whisking frequency is not always the same but might be varied from one whisking cycle to the next [6]. Most probably, both proper orientation and adapted active exploration are crucial for fine texture discrimination and thus complement the stereotyped active exploration that was investigated in this paper.

In addition to behavior exclusively based on whiskers, the robot is already equipped with an omnidirectional camera. This opens up the possibility of investigating behavior based on two different sensory modalities.

5 Conclusion

In this paper, we have shown that it is possible to classify tactile data of different textures acquired with artificial whiskers. In a first series of experiments, we have shown that four textures consisting of a smooth metallic surface, two different sandpapers and rough carton can be classified even when the position of the whiskers with respect to the texture is varied considerably. This result is a prerequisite for using the sensor on a robot without highly precise position control. In a second series of experiments, a mobile robot was used to acquire data in an open environment with walls of different

tactile quality. Here, the positions of the robot with respect to the wall were not specified but only limited loosely. Our experiments have shown that classification is not entirely reliable under real-world conditions. However, given sufficient data, a rough discrimination has been achieved. In the future we will use more biologically inspired sensory processing and sensory-motor feedback to refine the tactile capabilities.

Acknowledgements

This research has been supported by the IST-2000-28127 European project (AMOUSE). The natural rat whiskers were kindly provided by Mathew Diamond, SISSA, Cognitive Neuroscience sector, Trieste, Italy. The preamplifier board was built by Dirk Naumann, ATM Computing. Thanks to Simon Bovet for valuable discussions.

References

1. The Mathworks - Documentation on the Neural Network Toolbox.
2. E. Arabzadeh, S. Panzeri, and M.E. Diamond. Whisker vibration information carried by rat barrel cortex neurons. *Journal of Neuroscience*, 24(26):6011–6020, 2004.
3. E. Arabzadeh, R.S. Petersen, and M.E. Diamond. Encoding of whisker vibration by rat barrel cortex neurons: implications for texture discrimination. *Journal of Neuroscience*, 23(27):9146–9154, 2003.
4. M. Brecht, B. Preilowski, and M.M. Merzenich. Functional architecture of the mystacial vibrissae. *Behavioral Brain Research*, 84(1-2):81–97, 1997.
5. G.E. Carvell and D.J. Simons. Biometric analyses of vibrissal tactile discrimination in the rat. *Journal of Neuroscience*, 10(8):2638–2648, 1990.
6. G.E. Carvell and D.J. Simons. Task- and subject-related differences in sensorimotor behavior during active touch. *Somatosensory and motor research*, 12(1):1–9, 1995.
7. M. Diamond. personal communication. 2003.
8. M. Fend, S. Bovet, and V.V. Hafner. The artificial mouse - a robot with whiskers and vision. In *Proceedings of the International Symposium on Robotics, Paris*, 2004. on CD.
9. M. Fend, S. Bovet, H. Yokoi, and R. Pfeifer. An active artificial whisker array for texture discrimination. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.
10. M. Fend, H. Yokoi, and R. Pfeifer. Optimal morphology of a biologically-inspired whisker array on an obstacle-avoiding robot. In W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, editors, *Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life (ECAL)*, volume 2801 of *Lecture Notes in Artificial Intelligence*. Springer Verlag Berlin, Heidelberg, 2003.
11. E. Guic-Robles, C. Valdivieso, and G. Guajardo. Rats can learn a roughness discrimination using only their vibrissal system. *Behavioural Brain Research*, 31(3):285–289, 1989.
12. V.V. Hafner, M. Fend, P. König, and K.P. Körding. Predicting properties of the rat somatosensory system by sparse coding. *Neural Information Processing Letters and Reviews*, 4(1):11–18, 2004.
13. V.V. Hafner, M. Fend, M. Lungarella, R. Pfeifer, P. König, and K.P. Körding. Optimal coding for naturally occurring whisker deflections. *Proceedings of the 10th International Conference on Neural Information Processing (ICONIP), Istanbul, June 2003*, 2003.
14. J. Hipp. personal communication. 2005.

15. DaeEun Kim and Ralf Moeller. A biomimetic whisker for texture discrimination and distance estimation. In *Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior (SAB)*, pages 140–149, 2004.
16. M. Lungarella, V.V. Hafner, R. Pfeifer, and H. Yokoi. An artificial whisker sensor for robotics. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2931–2936, 2002.
17. S.B. Metha and D. Kleinfeld. Frisking the whiskers: Patterned sensory input in the rat vibrissa system. *Neuron*, 41:181–184, 2004.
18. C.I. Moore. Frequency-dependent processing in the vibrissa sensory system. *J. of Neurophysiology*, 91:2390–2399, 2004.
19. R. Andrew Russell and J. A. Wijaya. Object location and recognition using whisker sensors. *Australasian Conference on Robotics and Automation*, 2003.
20. A. Seth, J.L. McKinstry, G.M. Edelman, and J.L. Krichmar. Spatiotemporal processing of whisker input supports texture discrimination by a brain-based device. In S. Schaal, A. Ijspeert, A. Billard, S. Vijajamumar, J. Hallam, and J.-A. Meyer, editors, *Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior (SAB)*, pages 130–139. MIT Press, 2004.
21. J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593–600, Jun 1994.
22. J. A. Wijaya and R. Andrew Russell. Object exploration using whisker sensors. *Australasian Conference on Robotics and Automation*, 2002.

Analysing the Evolvability of Neural Network Agents Through Structural Mutations

Ehud Schlessinger[1], Peter J. Bentley[2], and R. Beau Lotto[1]

[1] Institute of Ophthalmology, University College London, 11-43 Bath Street, London EC1V 9EL
{e.schlessinger, lotto}@ucl.ac.uk
[2] Department of Computer Science, University College London, Malet Place, London WC1E 6BT
P.Bentley@cs.ucl.ac.uk

Abstract. This paper investigates evolvability of artificial neural networks within an artificial life environment. Five different structural mutations are investigated, including adaptive evolution, structure duplication, and incremental changes. The total evolvability indicator, E_{total}, and the evolvability function through time, are calculated in each instance, in addition to other functional attributes of the system. The results indicate that incremental modifications to networks, and incorporating an adaptive element into the evolution process itself, significantly increases neural network evolvability within open-ended artificial life simulations.

1 Introduction

Understanding the causal relationship between the functional structure of adaptive neural networks and ecological history is a fundamental objective in neuroscience research. And an important method for addressing this challenge is to model artificial neural networks within open-ended, ecologically relevant Artificial Life environments. The problem, however, is that standard neural network training algorithms (such as back-propagation) cannot be used in this paradigm, as such they do not offer the complete set of input and output values needed for training. An alternative is to therefore use the same mechanism nature does – evolution through 'natural selection'.

There are, however, many issues to consider when evolving neural networks. Fundamental to any of these is the evolution of network topology (i.e., modifications to its underlying node and connection structure). Here we address the question of the *process* by which network elements are to be added (and removed) by focusing, not on evolved network solutions as such, but on the evolvability of the systems itself.

Evolvability is the ability of a population to produce offspring fitter than any yet in existence [1], and not to produce less fit variants [13], and is therefore fundamental to the process of evolution itself. Evolvability is also known as evolutionary adaptability [8] and as such, a major element of evolvability is the capacity to adapt to changing environments by learning to exploit commonalities over time in those environments. By understanding evolvability and how to promote it, not only will it be possible to

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 312 – 321, 2005.

solve increasingly complex problems, but one may also better understand evolution of network systems generally.

The key properties required to generate systems exhibiting high evolvability are not well understood, particularly in the context of ecologically relevant artificial life simulations. Nonetheless, several factors are thought to be correlated with high evolvability.

(1) The mapping of genetic variation onto phenotypic variation [4, 15], and the selection of search operators used, determine the distribution of local optima in the search space, and affect search difficulty [1, 7]. More specifically, a many-to-one genotype-to-phenotype mapping (a redundant mapping), is essential for evolvability. By enabling some mutations to be phenotypically irrelevant, it is possible to better explore the search space through neutral networks [5]. Evolution of neural networks, in our view, particularly those that are used for control and classification, qualifies for the complex mapping condition; Fogel [6] defined an evolved neural network's phenotype as its behaviour, and not its constituent weights. Using this definition, changing many aspects of a neural network would not necessarily change its phenotype (behaviour).
(2) Gradual effects of the search operators seem to play an important part [2, 9].
(3) Structural duplication and modularity are recognised as promoters of evolvability [17], as they enable evolution to 'reuse' structures within networks [10].
(4) Finally, the ability of evolution of adapting elements of itself can also promote evolvability, since it enables evolution to differentially tune search operators throughout evolution [4, 7].

This paper analyses the effect on the evolvability of an artificial life simulation, as measured by the evolvability indicator, E_{total}, using five different types of structural mutations. Each of the mutation types incorporates the various principles described above for increasing evolvability. Secondary effects on evolved traits were also measured, such as the number of successful 'runs', quality of evolved solutions, and the variability of the evolved forms.

2 System

Mosaic World is an A-Life system designed for exploring the computational principles by which vision can overcome stimulus ambiguity [12]. Mosaic World offers a virtual environment made up of a 2D grid of 'coloured' surfaces under multiple simulated light sources. This environment emulates key characteristics of natural scenes. The space is inhabited by virtual agents, 'critters', that survive by consuming positive resources and avoiding negative resources. Every surface's value is determined from its reflectance – its colour. Consumed resources slowly regenerate.

The critter population is maintained by the critter reproduction. Critters can reproduce both sexually and asexually. In the event that all critters perish, a new population is created, where 80% are random critters and the rest are mutated clones of critters that showed general promising survival skills earlier in the run. Every critter starts out with a certain amount of energy, and dies if it runs out of energy. Critters slowly lose

energy over time, or due to moving, turning, resource consumption and reproducing. A critter dies if it steps over the edge of the world, or into a hole.

Critter behaviour (such as mating, eating, and moving) is determined by the output of a modified 3D feed-forward neural network. Network topology is determined by the critter's genome, though its behaviour is an emergent property of the interaction between the nodes within this topology. The input layer contains receptors (which are input units modified to enable evolution of vision) and a health monitor unit, which receives the percentage of the critter's remaining health. The hidden layer contains standard hidden units. The output layer contains standard output units, which determine the critter's behaviour. Every unit in the network has an [x,y] coordinate relative to the critter's centre, which defines its location in its layer. Using the layer and the [x,y] coordinate, networks of different architectures can be crossed over during sexual reproduction, as each network possesses the same coordinate reference frame.

The units of the network communicate through connections that can extend between units from higher layers to lower layers, and can also connect units to empty coordinates in the network (unconnected connections). Connections can be active or inactive. Only active connections participate in the feed-forward process.

2.1 Mutation Operators

In this work, we focus on an investigation of evolvability using several types of structural mutations. A full description of the non-structural mutations is in [12].

For a mutation type to be useable, it must have the ability to completely alter a neural network's structure by adding and deleting elements. In order that we are able to test the effects of suggested principles thought to increase evolvability, every mutation type used in our experiments incorporated some of these principles. The three principles tested are: *incremental changes to network topology*, where every change done to the network structure is very small, *adaptive evolution,* where evolution can modify some aspects of itself, and *structural duplication*, where existing substructures of the network are copied and can be reused.

Deletion and addition of units (receptors, hidden) are performed using *Delete Unit* (0.5% per unit) and *Add Unit* mutations (2%). Deletion and addition of connection weights are performed using *Delete Connection* (0.1% per connection) and *Add Connection* mutations (1%). When a unit is added, it is randomly placed in the appropriate layer with a bias towards the centre and forms connections with units in the adjacent layers. All new connections are initialised with random values. When a unit is removed, all its outgoing connections are removed. If, as a result of a unit being deleted, a connection now has no end destination, it remains in the network. These connections are termed 'unconnected connections', and as such are not used in feed-forward processing, only becoming functional again if the old unit is replaced.

Receptors in the input layer can change locations through *Drift* mutation (0.3% per receptor). *Switch* mutations (0.3% per element) can cause a connection or a receptor to become active or inactive. An inactive element is not used in the feed-forward process, and is deleted from the genome when a number of generations have not activated it again.

The above probabilities were empirically determined to be suitable during the course of 15 experiments (roughly 750 runs).

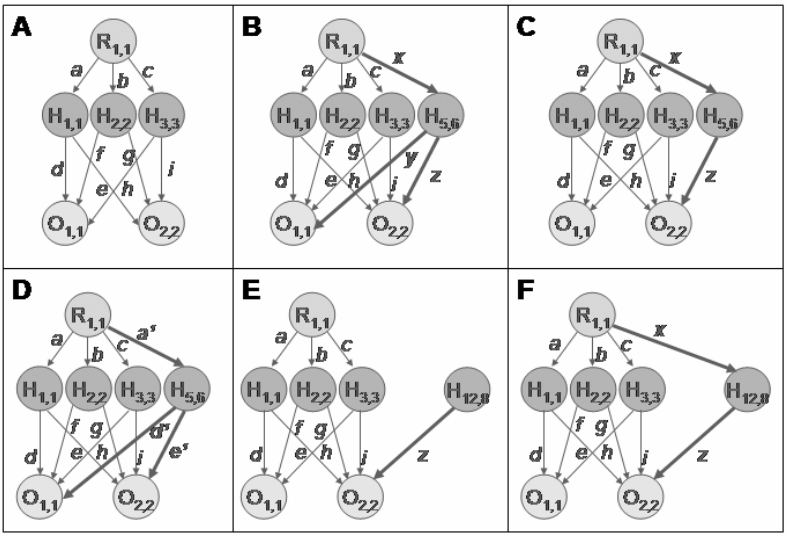


Fig. 1. Illustrating addition of a hidden unit using the five types of mutations. [A] The original neural network with 1 receptor, 3 hidden units, and 2 output units. [B] Using mutation type (i), new unit (H5,6) is fully connected through 3 random connections. [C] Using mutation type (ii), new unit (H5,6) connects to (R1,1) and (O2,2). [D] Using mutation type (iii), new unit (H5,6) is a clone of (H1,1). [E] Using mutation type (iv) new unit (H12,8) only connects to (O2,2) as the distance parameter is very high. [F] Using mutation type (v) new unit (H12,8) connects to the closest receptor (R1,1) and closest output unit (O2,2).

The following types of structural mutations were used in the experiments (see fig. 1). The probabilities of these mutations occurring are identical for all types. The tested principle appears in parenthesis.

Type 1 - fully connected (non-gradual changes): New units connect to **all** units in adjacent layers. Using this mechanism, every mutation makes a large change to the networks.

Type 2 - single connection (gradual changes): New units connect to a **single**, randomly chosen, unit in every adjacent layer. The *Delete Unit* mechanism is disabled – units are automatically removed when they have no outgoing or no incoming connections. Using this mechanism, every mutation makes a small change to the network.

Type 3 – reuse of structures (structural duplication): Added units are cloned from a random unit in the same layer. The new unit possesses a copy of every incoming and outgoing connection of the original.

Type 4 – distance dependent (adaptive evolution, gradual changes): Added units connect to **all** units in adjacent layers within a given distance. The distance parameter is an evolvable gene of a critter. By evolving a low distance parameter, the change to the network can be very small or very large.

Type 5 – shortest connection (adaptive evolution, gradual changes): Added units connect to the **closest** unit in every adjacent layer. The *Delete Unit* mechanism is disabled – units are automatically removed when they have no outgoing or no incom-

ing connections. Using this mechanism, every mutation makes a small change to the network. Additionally, evolution can now utilise the 3D coordinate system to create modules, which adds an adaptive element (albeit weaker than type 2).

2.2 Measuring Evolvability in Mosaic World

Mosaic World is more than just a population of individual critters – it is a dynamic ecosystem in which critters survive if their genomes enable them to interact with each other and their current environment effectively enough to gather resources [12].

Previously suggested measurements of evolvability [1, 13] do not take into account conditions specific to the ecologically relevant conditions of Mosaic World, and as a result they could not be used. These methods require accurately measuring fitness, which is not feasible for three reasons: First, no one statistic encapsulates all the required behaviours a critter must know to be termed fit. Second, the fitness of all critters is linked, as critters compete against each other on resources; a fit critter, effectively, decreases the fitness of other critters. Third, although reproduction does not directly contribute to a critter's fitness, controlling reproduction is crucial to the species' collective fitness: The population, as a whole, must replenish itself at a rate that is sustainable by the available resources of the world. Thus, a critter must share some of this collective fitness.

Therefore, the evolvability measurement we use here is based on the evolvability used in the Avida ALife environment [11]. This measurement was expanded by factoring environment difficulty. We believe that evolvability can either be expressed by demonstrating that a population gradually improves over time, or alternatively, by showing a population adapting to an environment that gradually becomes more challenging. By quantifying these aspects, we define the total evolvability indicator in Mosaic World, E_{total}, using equation (1) – its range of possible values is 0 to 1, and the *evolvability function through time*, using equation (2). Both measures incorporate four different elements: survivability, population success, environment difficulty and time variance.

Survivability: The critter's survival ability is best expressed by its age. A critter that can survive for long obviously managed to learn important skills required to survive in the world. Furthermore, by surviving longer, a critter may get more opportunities to reproduce and as a result spread fit genetic material to its offspring.

Population success: A population's 'fitness' is best expressed by its size at a given time. A population that managed to maintain itself through time, collectively learned how to balance resource consumption and reproduction through its constituent critters. Also, a larger population has more individuals that pass on traits to offspring, and is more likely to survive a 'catastrophe' purely because of its greater size.

World difficulty: In many experiments the environment is altered over time to make it more challenging for a critter to survive. A population that manages to survive under conditions in which the selection pressure continuously grows, shows an indication of adaptability, and thus, evolvability. This aspect of the equation is controllable by the researcher and must be directly tied in, from a numerical point of view, to the difficulty of the world in order to measure evolvability, i.e. if survival in the world at time t is twice as hard as the initial conditions, the difficulty factor at time t is 2.

Time: Only by looking at the relative changes of survivability, population success and world difficulty over time, we can precisely obtain the total evolvability measure.

In conclusion, these four elements measure the capacity of Mosaic World's population to evolve. A population that maintains large numbers, where each agent survives for long, in an increasingly difficult environment, consistently through time – can be said to be a population with a great capacity to evolve. Therefore, this function measures the capacity of a population to generate fit offspring through time.

$$E(t) = \frac{D(t)}{D_{\max}} \frac{\sum_{i=0}^{P_t} \left(\frac{A_{t,i}}{A_{\max}} \right)}{P_{\max}} \quad (1)$$

$$E_{total} = \frac{\sum_{i=0}^{t} E(t)}{t_{\max}} \quad (2)$$

$$Resilience = \frac{\sum_{i=0}^{t} iE(i) - n\overline{E(t)}\bar{t}}{\sum_{i=0}^{t} i^2 - n(\bar{t})^2} \quad (3)$$

$$Stamina = \overline{E(t)} - \text{Resilience}\bar{t} \quad (4)$$

Where: E_{total} is a population's evolvability indicator, E(t) is the evolvability at time t, D(t) is the difficulty factor at time t, D_{max} is the maximal difficulty of D(t), P_t is the size of the population at time t, $A_{t,p}$ is the age of a member of population p at time t, A_{max} is the critter maximum life span, P_{max} is the maximal population the environment can support, t_{max} is the total length of time of the experiment, n is the number of data values.

Example: With a population size *P* of 400 at time 10000, all critter ages *A* are 1500, the difficulty factor *D* at time 10000 is 100, using maximum difficulty D_{max} of 350, maximum population size P_{max} of 10000, and maximum age A_{max} of 15000, evolvability at time 10000 is E(10000)=100/350*(400*1500/15000)/10000=0.00114.

By extracting the height and the slope of a linear trendline of the evolvability function through time (using equations (3) and (4)), we gain two extra indicators: (i) *Resilience (slope)*: Defines the resilience of the population to change. Lower values indicate populations more tolerant to change. (ii) *Stamina (height)*: Defines the population's ability to thrive when conditions are easy.

3 Experiments

The main objective of the experiments was to measure the evolvability function through time, E(t), and the total evolvability, E_{total}. A secondary objective was to obtain additional statistics examining effects other than evolvability of the structural mutations used: Variability of evolved forms (average structure), quality of critter solutions and the percentage of successful runs (a run failed when no population of critters evolved without the need for a restart).

To this end, two sets of experiments were performed. Each of the experiments required multiple populations that were evolved using the five structural mutations. Therefore, at least eight successfully evolved populations were collected for each of the mutation types (using the same randomly generated world). Each run started with identical population characteristics (all critters possessing fully connected networks: 3 receptors, 3 hidden units and 8 output units, 33 connections), and was stopped after 550,000 time steps. During each run, the regeneration rate of consumed surfaces was slowly reduced to increase challenge and force critter populations to adapt. Initially, consumed surfaces regenerated every 13 time steps 3% of their maximal value. Every 3,500 time steps regeneration slowed down by one unit, until the regeneration rate of 99 was reached. To analyse the effects of the mutation operators only, crossover was disabled during all runs and experiments.

Experiment 1 - Measuring Evolvability through Adaptation: This experiment attempted to test the maximum difficulty that a population can adapt to. Using the collected data and equations (1) and (2), E(t) was charted and E_{total} was calculated. Since the regeneration rate has a direct effect on the difficulty of the world, the rate was used as the difficulty factor in equation (1). Therefore, five copies of the five longest-lived critters of every evolved population were placed in an identical test world. The starting regeneration rate was set to 99, and every 1,000 time steps the it slowed down by one unit, indefinitely. A run was finished when all critters died.

Experiment 2 - Measuring the Quality of Evolved Solutions. This experiment attempted to measure the quality of evolved solutions, the critters. The criterion used was critter survivability, which was measured by averaging the critter survival ages across runs. To do this accurately, the effect of the critters on each other was negated by prohibiting reproduction, and by placing a very small number of critters in every world. Furthermore, the difficulty of the world was made static by fixing the regeneration rate (to 99). Therefore, five copies of the five longest lived critters of every run were placed in an identical test world. Critters were expected to survive as long as they could. All runs were stopped after 10,000 time steps, and were repeated 3 times. Critters that survived until the end of the run were assumed to have died then.

4 Results

In table 1, we see E_{total} for each type (as a percentage of the maximum E_{total} of type 4), and the resilience and stamina for each type (using equations (3) and (4) and divided by type 4's resilience for comparison purposes). In fig. 2, we see the evolvability function (weighted average) through time with E_{total} appearing in the legends for every type. Table 2 shows the minimum, maximum and average of the maximum regeneration rate a population could adapt to and of critter average survival age, as well as the percentage of successful runs and the average critter structure per type.

When comparing the E_{total} of all types, it is clear that adaptive evolution and gradual changes to networks increase E_{total}, whereas non-gradual changes, and structural duplication decrease it. Types 4 and 5, both utilising adaptive evolution and gradual

Table 1. The evolvability elements incorporated, the obtained E_{total} (as a percentage of E_{total} of type 4) and the extracted resilience and stamina values using a linear trendline of E(t) for every type (divided by type 4's resilience for comparison purposes)

Mutation type	Element Incorporated	E_{total} (%)	Resilience	Stamina
4	Adaptive evolution, Gradual changes	100.00%	-1	5.68
5	Adaptive evolution, Gradual changes	98.12%	-1.13	6.39
2	Gradual changes	78.50%	-0.98	5.53
1	Non-gradual changes	71.47%	-0.94	5.29
3	Structural duplication	41.58%	-0.41	2.34

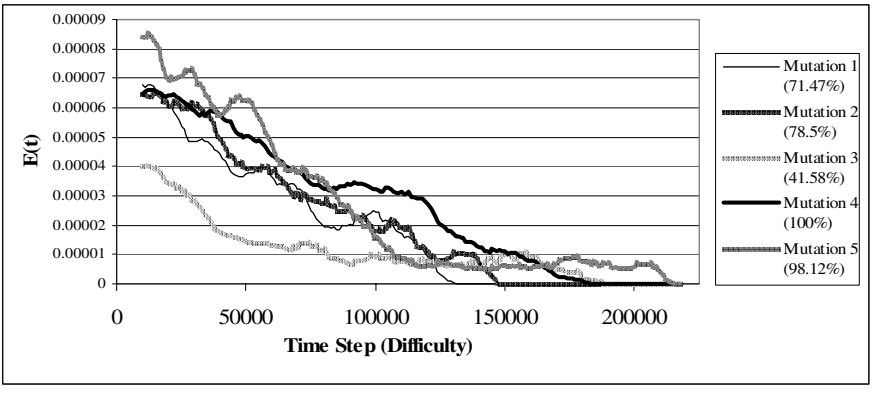


Fig. 2. The evolvability function (weighted average) for the five types of structural mutations and their relative evolvability indicator (of E_{total} for mutation type 4)

Table 2. Several statistics (average, min, max) describing the maximum regeneration rates the tested populations adapted to and the critter survivability, in addition to the average critter structure, and percentage of successful runs; broken down according to mutation types

Mutation type	Maximum adapted regeneration rate: Ave. (Min.-Max)	Survival age: Ave. (Min.-Max.)	Ave. critter structure: Receptors, Hidden (Connections)	Successful runs (%)
Random Critter		57.36 (56.08-59.48)	3, 3 (33)	
1	191.14 (119–222)	3182.37 (1277.23-4600.12)	4.03, 3.13 (29.47)	64%
2	197.12 (159–237)	3733.34 (2781.13-4801.6)	8.32, 10.74 (108.70)	73%
3	163.87 (109–277)	2388.49 (893.44-5339.6)	4.86, 4.51 (41.45)	50%
4	224.36 (171–272)	3905.31 (1625.16-5021.96)	4.98, 6.26 (55.48)	69%
5	202.62 (167–305)	3651.06 (2613.92-5321.28)	10.39, 12.21 (144.25)	62%

changes, had the highest E_{total} with type 4 the higher of the two. The difference in their evolvability functions were, however, significantly different: Type 5 had – on average – a higher stamina, but it was less resilient than type 4, and its populations quickly weakened as difficulty increases. Type 4 was more resilient, as evident in its average adaptation rate. Overall, the data suggests that the Type 4 structural mutation is slightly more evolvable [note that Type 4's average survival age was also the best of all runs; Type 5's was lower, but still very good]. It could be said, however, that Type 5, having a higher stamina, and lasting the longest in our adaptation experiment, is the most evolvable type. However, we believe the total area under the curve is the best indication of evolvability, since this measure takes into account both stamina and resilience.

Type 2, causing only small increments to the network, had a higher E_{total} than the Type 1's. It also had the best average survival age, and best rates of success. Despite its populations' decent performance, once the environment becomes too challenging, however, it its evolvability decreases significantly, causing its populations to perish.

Type 1, causing large changes to the network, had mediocre statistics and a low E_{total}. Generally, it seemed unable to utilise the structural mutations: on average, only one receptor, and no hidden units, were added at all. We believe this is another gauge of its low evolvability.

Type 3, utilising structural duplication, had the lowest E_{total} as well as the lowest scores on all other tests. It would be easy to dismiss this method of evolution as completely non evolvable, except for the fact that, despite having the low results of the vast majority of type 3's runs, some of its individual runs scored the *highest* average survival age and the near highest adaptation rates. The weakness of this approach is that cloning a fully connected hidden unit usually results in very large changes to the network (in some instances, 10+ connections being added at once), so it is possible this negative evolvability promoter far outweighs the positive evolvability gained by the structural duplication aspect. We can only deduce that this method has potential, but its weakness far outweighs its strength.

Looking at the evolved forms, it is obvious that all types utilised the structural mutations to increase their network's complexity, with some more than others. Some types in particular (types 2, 5) resulted in networks significantly larger than the starting networks. However, it does not seem as if the larger networks were inherently better or worse than the smaller ones. Interestingly, it seems as if these larger networks tended to provide the most consistent critters in terms of average survival age.

A possible criticism would suggest that highly evolvable populations would continue evolving forever, with E(t) values always above zero and E_{total} tending to infinity. However, in our system this is impossible. At the slowest rates of regeneration tested in our experiments, there are not enough resources left to support individuals, regardless of their genomes. Inevitably, evolvability must drop to zero at some point, for there will be no critters left in the population to evolve. Such eventual resource limitation leading to extinction is inevitable in all real and modeled systems (time will always be limited, if nothing else), so an infinite E_{total} may be impossible to achieve.

5 Conclusions

The aim of this study was to investigate the evolvability of neural networks within an artificial life simulation. Specifically, we tested the efficacy of five different types of

structural mutations, which incorporate different general principles thought to be important for network evolvability. Two experiments were performed, and the resulting E_{total} and evolvability function over time were calculated and compared.The experiments conducted indicate that certain principles increase evolvability when used to evolve neural network artificial agents. The two most significant promoters of evolvability are adaptive evolution and gradual changes to the networks. Structural duplication, despite exhibiting on average very low evolvability, showed some potential by evolving some of the best individual runs. Non-gradual changes to the networks seemed to be detrimental to evolvability (or at least, did not seem to increase it).

To summarise: the method chosen to in evolving neural networks for artificial life simulations plays a significant factor in all elements of the evolved runs. Researchers attempting to evolve neural networks are encouraged to use these principles.

References

1. Altenberg, L. (1994). The Evolution of Evolvability in Genetic Programming. In: *Advances in Genetic Programming*, K. E. Kinnear Jr., ed. MIT Press.
2. Altenberg, L. (1995). Genome growth and the evolution of the genotype-phenotype map. in W. Banzhaf and F. H. Eeckman, eds, *Evolution and Biocomputation: Computational Models of Evolution*. 205-259. New York. Springer-Verlag, Berlin, Heidelberg, 1995.
3. Astor, J. and Adami, C. (1998). Development and evolution of neural networks in an artificial chemistry. Proc. *Third German Workshop on Artificial Life*, Verlag Deutsch, 15-30
4. Bedau, M. A. and Packard, N.H. (2003). Evolution of evolvability via adaptation of mutation rates. *Biosystems* 69(2-3): 143-162
5. Ebner, M., Shackleton, M. and Shipman, R. (2001). How neutral networks influence evolvability. *Complexity*, 7(2) 19-33, Wiley Periodicals, 2001.
6. Fogel, D.B. (1995). Phenotypes, Genotypes, and Operators in Evolutionary Computation. Proceedings of the *1995 IEEE International Conference on Evolutionary Computation, Perth*, Australia, IEEE Press, 193-198.
7. Glickman, M. and Sycara, K. (1999). Comparing Mechanisms for Evolving Evolvability. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 1999 Workshop Program*. Orlando, Florida, USA, July 13, 1999.
8. Kirschner, M. and Gerhart, J. (1998). Evolvability. *Proceedings of the National Academy of Sciences*, 95:8420-8427.
9. Kumar, S. and Bentley, P. J.(2000). Implicit Evolvability: An Investigation into the Evolvability of an Embryogeny. A late-breaking paper in the *Genetic and Evolutionary Computation Conference* (GECCO 2000), July 8-12, 2000, Las Vegas, Nevada, USA.
10. Nolfi, S. and Parisi, D. (2002). Evolution of artificial neural networks. In *Handbook of brain theory and neural networks*, Second Edition. Cambridge, MA: MIT Press, 418-421.
11. Ofria, C., Adami, C., and Collier, T. C. (2002) Design of Evolvable Computer Languages. IEEE Transactions on Evolutionary Computation, 6:420-424.
12. Schlessinger, E., Bentley P.J. and Lotto R.B. (2005) Evolving Visually Guided Agents in an Ambiguous Virtual World. To appear in the *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2005*. Washington, DC, USA, June 25-29, 2005.
13. Smith, T.M.C., Husbands, P., Layzell, P. and O'Shea, M. (2002). Fitness Landscapes and Evolvability. *Evolutionary Computation*, 10(1):1-34.
14. Stanley, K.O. and Miikkulainen, R. (2002). Evolving Neural Networks Through Augmenting Topologies. *Evolutionary Computation* 10(2):99-127.
15. Wagner, P. W. and Altenberg, L. (1996). Complex adaptations and the evolution of evolvability. *Evolution* 50, 967-976.

Coevolutionary Species Adaptation Genetic Algorithms: A Continuing SAGA on Coupled Fitness Landscapes

Larry Bull

School of Computer Science,
Faculty of Computing, Engineering & Mathematical Sciences,
University of the West of England,
Bristol BS16 1QY, U.K
larry.bull@uwe.ac.uk

Abstract. The Species Adaptation Genetic Algorithm (SAGA) was introduced to facilitate the open-ended evolution of artificial systems. The approach enables genotypes to increase in length through appropriate mutation operators and has been successfully exploited in the production of artificial neural networks in particular. Most recently, this has been undertaken within coevolutionary or multi-agent scenarios. This paper uses an abstract model of coevolution to examine the behaviour of SAGA on fitness landscapes which are coupled to those of other evolving entities to varying degrees. Results indicate that the basic dynamics of SAGA remain unchanged but that the rate of genome growth is affected by the degree of coevolutionary interdependence between the entities.

1 Introduction

Harvey [8] introduced a form of Genetic Algorithm (GA)[10] which is extended to permit genomes of varying length with the aim of enabling open-ended evolution - the Species Adaptation Genetic Algorithm (SAGA). Related work on using variable length genomes primarily relies upon recombination to adjust the size of offspring, such as Pittsburgh-style Learning Classifier Systems [20] and Genetic Programming [16]. Using a version of the abstract *NK* fitness landscape model [13], Harvey showed, by including a bias, that gradual growth through small increases in genome length via mutation is sustainable whereas large increases in genome length per growth event is not sustainable. This is explained as being due to the fact that a degree of correlation between the smaller fitness landscape and the larger one must be maintained; a fit solution in the former space must achieve a suitable level of fitness in the latter to survive into succeeding generations. Kauffman and Levin [13] discussed this general concept with respect to fixed-size fitness landscapes and varying mutation step sizes therein. They showed how for long jump adaptations, i.e., mutation steps of a size which go beyond the correlation length of a given fitness landscape, the time taken to find fitter variants doubles per generation. Harvey's [8] growth operator is a form of mutation which adds g random genes to an original genome of length G. Hence he draws a direct analogy between the size of g and the length of a jump in a traditional landscape; the larger g, the less correlated the two landscapes will be regardless of the underlying degree of correlation of each. SAGA has since been used effectively in numerous evolutionary robotics experiments [e.g., 9].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 322 – 331, 2005.

Prior to Harvey's work, Lindgren [17] had presented the use of "duplication" and "split" mutations within a version of the Iterated Prisoner's Dilemma (IPD) game [1]. These operators doubled by copying or halved the length of a player's strategy respectively and Lindgren [17] showed sustained growth consistently emerged. Here, as pointed out by Lindgren, a doubling event has no effect on the strategy/phenotype of an individual and so such mutations are effectively neutral. It is later, through standard gene mutations, that more complex strategies can develop. This process of gene duplication followed by divergence has been highlighted as a factor in the evolution of complexity within natural systems [e.g., 18]. It can also be noted that the IPD is coevolutionary in nature - the fitness of individuals is dependent upon the environment in which they exist.

Bull et al. [6][5] used the coevolutionary version of the *NK* model, the *NKCS* model [14], to examine symbiogenesis [e.g., 15], the process which causes an increase in complexity by the bringing together of genomes from separate species. The model allows systematic adjustment of the degree of interdependence between coevolving species; fitness landscapes are coupled and hence the adaptive moves of one species changes the shape of the fitness landscapes of the others. Bull et al. show how, for significant degrees of interdependence, it is more effective for species to become genetically linked, here to double their genome lengths, than stay separated. That is, by becoming genetically merged, a species no longer suffers the changes in their fitness landscape caused by their partner. Symbiogenesis has been a fundamental evolutionary process in nature [e.g., 18] and has been exploited within artificial systems [e.g., 22].

Hence the results of Lindgren [17] and Bull et al. [6][5] indicate that large increases in genome length are beneficial under certain circumstances within a coevolutionary scenario. Harvey's [8] findings of short growth only being sustainable were made on stationary fitness landscapes. More recently, SAGA has been successfully used to develop the neural controllers for mobile robots which exist within a multi-agent team [e.g., 19], i.e., for systems which are coevolutionary in nature. A SAGA-like approach has also been used to develop networks within ensembles of controllers for mobile robots, in a version of Holland's [11] Learning Classifier System architecture [12], again, a coevolutionary approach. In this paper, a version of the *NKCS* model is used to re-examine the behaviour of SAGA within a coevolutionary context. In particular, the effects of varying the degree of interdependence between species on the rate of growth are examined, along with other aspects of the basic approach.

The paper is arranged as follows: the next section introduces the *NKCS* model. Section 3 describes the implementation of SAGA used. Section 4 shows how the degree of interdependence effects the rate of growth. Section 5 examines the effects of increasing the size of the increments in length. Section 6 shows how the frequency of growth events can be self-adapted. Finally, all results are discussed.

2 The *NKCS* Model of Coevolution

Kauffman and Johnsen [14] introduced the *NKCS* model to allow the systematic study of various aspects of ecological evolution. In the model an individual is represented by a haploid genome of N (binary) genes, each of which depends upon K other genes in its genome. Thus increasing K, with respect to N, increases the epistatic linkage. This increases the ruggedness of the fitness landscapes by increasing the number of fitness

peaks, which in turn increases the steepness of their sides and decreases their typical heights. Each gene is also said to depend upon C traits in the S other species with which it interacts. The adaptive moves by one organism may deform the fitness landscape(s) of its partner(s). Altering C, with respect to N, changes how dramatically adaptive moves by each species deforms the landscape(s) of its partner(s). It is shown that as C increases, mean performance drops and the time taken to reach an equilibrium point increases, along with an associated decrease in the equilibrium fitness level.

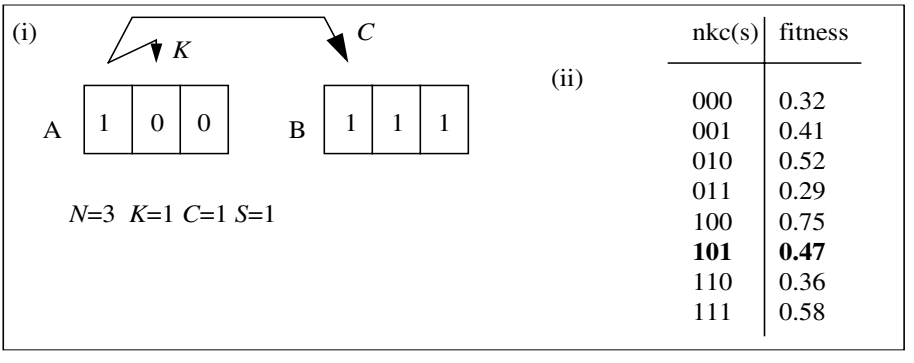


Fig. 1. Showing an example *NKCS* function. (i) Shows each gene depends on one gene locally (K) and one gene (C) in one other genome (S). Therefore there are eight possible allele configurations, each of which is assigned a random fitness, as shown in (ii). Total fitness is the normalised sum of these values.

The model assumes all intergenome (C) and intragenome (K) interactions are so complex that it is only appropriate to assign random values to their effects on fitness. Therefore for each of the possible K+CxS interactions, a table of $2^{(K+1+C\text{x}S)}$ fitnesses is created, with all entries in the range 0.0 to 1.0, such that there is one fitness value for each combination of traits (Figure 1). The fitness contribution of each gene is found from its individual table. These fitnesses are then summed and normalised by N to give the selective fitness of the total genome (the reader is referred to [14] for full details of the model).

Kauffman and Johnsen used populations of one individual (said to represent a converged species) and mutation to evolve them asynchronously (e.g., see also [7]). In this paper a steady state genetic algorithm is applied to a population-based version of the model. As well as the aforementioned work on symbiogenesis, GAs have previously been used with the *NKCS* model to examine multicellularity [e.g., 2] and eusociality [e.g., 4], as well as aspects of coevolutionary optimization [e.g., 5].

3 A Coevolutionary SAGA

A standard steady state GA is used here. Selection for reproduction uses the traditional fitness proportional scheme, picking one parent from the population of size P. Offspring are created by applying a standard gene mutation operator at rate $p_m = 1/N_0$, where N_0

is the initial length of all genomes in the population. At a rate p_g offspring have g random genes added to the right-hand end of their genome. Similarly, at a rate p_c offspring have g random genes cut from the right-hand end of their genome if the resulting genome will remain greater than or equal to N_0. Offspring replace an individual from within the population selected inversely proportional to fitness.

The *NKCS* model used contains two species: the evolving species as described and another said to represent all other aspects of the species' environment (E). To avoid the usual equilibria experienced within the model [14], E is represented by a single individual of length N_0 which randomly alters m_E genes at rate p_E per generation of the steady state GA. The fitness of all evolving individuals is recalculated upon a change in the environment. In this way, the behaviour of the SAGA in an environment of perpetual novelty can be examined.

As in [8], genomes are assumed to be in a ring and the K genes are those which surround the given gene (symmetrically), as opposed to being randomly chosen throughout the genome. The C genes within the environment are chosen randomly for each gene. Fitness tables are created for new genes as they are added under the SAGA up to a maximum allowed size of 500. In this way the addition or removal of genes results in disruption to the "end" genes of the genome only.

To create a selective pressure for growth, fitness values are normalized by the average length of the genomes within the population (N_P) rather than N_i as described above. Harvey used (N_i / (N_i +N_P)) but experimentation here has found the simpler metric sufficient to create a selective bias whilst also tending to maintain fitnesses in the range [0,1] as in the original models.

Unless otherwise stated, all runs reported are the average of 10 runs on 10 *NKCS* landscapes (i.e., 100 runs) with P=100, $N_0 = 16$, $p_m = 1/N_0$, $p_g = p_c = 0.01$, $p_E = 0.01$, $m_E = 1$, $g = 1$.

In the following sections this model is used to examine the effects of varying the degree of interdependence between the species under evolution via SAGA and the rest of its environment for various parameter settings.

4 Low Growth: $g = 1$

Figure 2 shows the behaviour of the SAGA population for various values of K and C. As can be seen, for a given value of underlying epistasis, increasing the degree of interdependence increases the rate of genome growth maintained within the population. This is true for K=0, i.e., the case where all genes in a genome are independent and hence the fitness landscape contains a single unimodal peak, as well as for increased amounts of landscape ruggedness: $0<K<4$ are known to represent correlated rugged landscapes with their being completely uncorrelated thereafter [21].

Hence, as in the aforementioned work of Lindgren [17] and Bull et al. [6][5], the coevolutionary scenario can be conducive to greater genome growth. Kauffman and Johnsen [14] highlighted how the higher the degree of interdependence C, the lower the typical mean fitness before an equilibrium is reached, explained as being due to the degree of landscape movement caused by changes in the environment.

This is seen in Figure 2, using an evolving population rather than a genetic hill-climber, in that the difference between mean fitnesses is typically greater for higher C compared

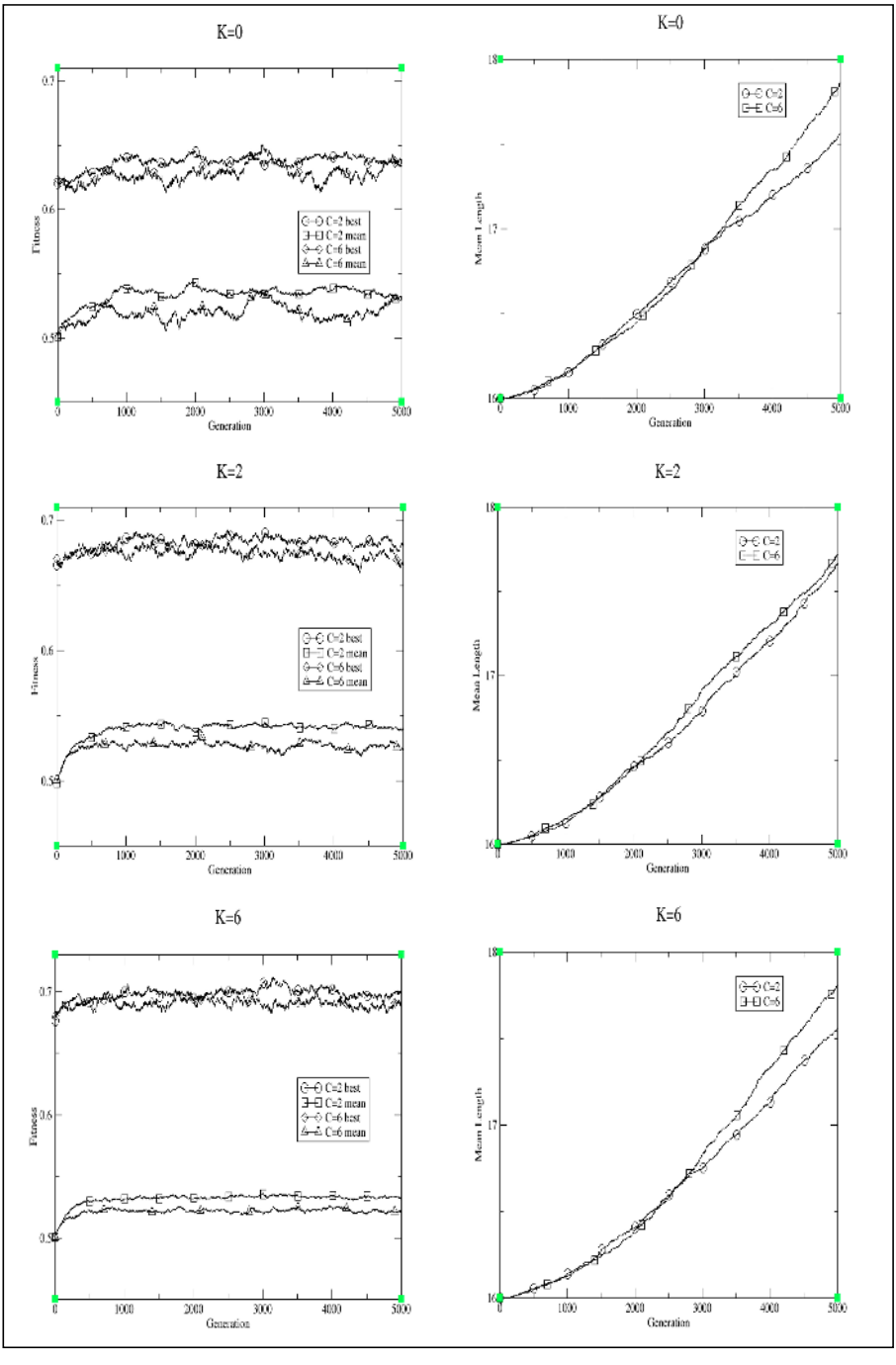


Fig. 2. Fitness and growth for various K and C

to the differences in best fitnesses. As noted above, for a genome of increased size to propagate within a population it must obtain a fitness which is comparable to those of shorter length. In the higher C case, the mean fitness of the population is lower and this reduces the selection pressure on the added random genes to make a significant contribution to fitness and hence allows for more growth. Similarly, results (not shown) from increasing the frequency of change in the environment, e.g., $p_E = 0.1$, find mean fitnesses are further decreased, enabling a higher rate of growth.

5 Higher Growth: $g > 1$

Figure 3 shows the behaviour of the same SAGA as in Figure 2 but with two, four or six genes added (or removed) per growth (or cutting) event. As can be seen, in all cases the extent of growth is greater but the fitnesses achieved are, over the longer term, lower than those in Figure 2. This was true for all K tried (not shown).This finding corresponds to the general behaviour observed by Harvey [8] using a different form of SAGA, as discussed above, in that large jumps in the space of possible genome lengths are disruptive to the evolutionary process since the two spaces are unlikely to be correlated to any significant degree; the effect appears to be an underlying feature of such open-ended artificial evolutionary systems. Here the fitnesses rise rapidly and then fall to a lower value than those previously obtained for lower g as the population becomes over-run with maladapted but longer genomes. That is, the bias in the normalization process used here encourages growth but at the expense of fitness in the longer term.

As in Harvey's work [ibid.], the initial genomes were of length sixteen (N_0=16). Figure 4 shows the effects of various g for a larger initial genome N_0=64. As can be seen, g=4 is not disruptive to the evolutionary process and fitnesses are maintained as growth continues. Results for larger amounts of growth, e.g., g=6, find that, again, fitnesses begin to suffer over the longer term. Therefore the extent of genome growth which can be sustained is proportional to the length of the genome; the larger the original genome, the larger g can be before the two landscapes are sufficiently uncorrelated. Results suggest that roughly $g = N_0/16$ is appropriate here.

The effects of altering the frequency at which growth/cut events occur have also been investigated. With g=1 but $p_g = p_c = 0.1$, as opposed to 0.01 above, the same general behaviour is seen as in Figure 3 (not shown). That is, sustainable growth is not only dependent upon the size of the jump but also the frequency at which it occurs; many small increases are as disruptive as less frequent large increases.

6 Self-adaptive Growth

Hurst and Bull [12] have introduced the use of self-adaptation within the SAGA. In particular, they allow the frequency of growth/pruning events, with a fixed size of g=1, to be self-adapted by each individual within the population. Here each solution effectively carries its own p_g and p_c and these probabilities are tested on the production of offspring. The probabilities have mutation applied to them beforehand, i.e., along with the functional genes, where they are adjusted using a Gaussian distribution N(0,1). Figure 5 shows the results of seeding a single parameter to control both the growth and cutting probabilities uniform randomly in the range [0, 0.02], i.e., such that the mean is

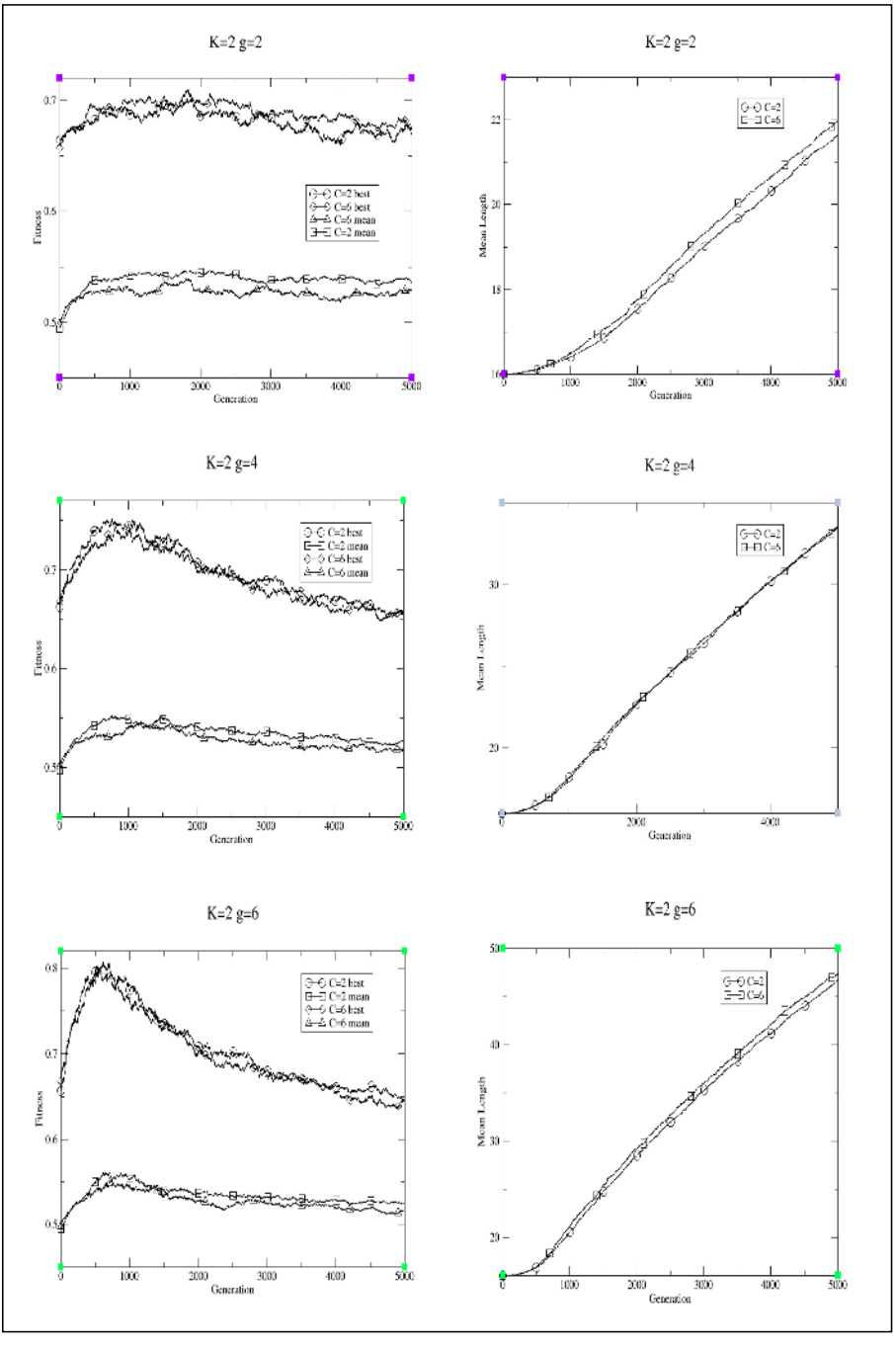


Fig. 3. Fitness and growth for K=2 and various C with larger g

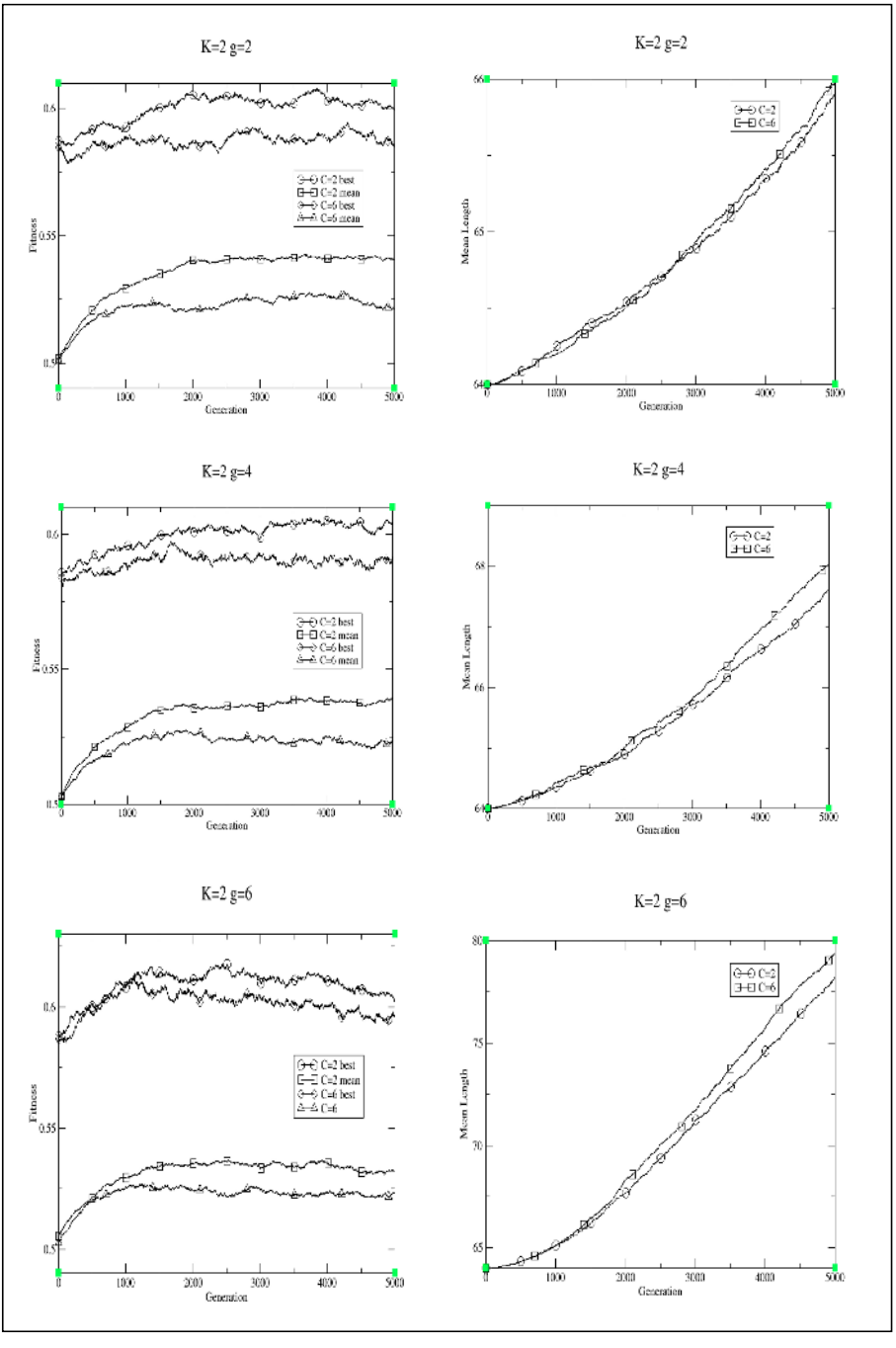


Fig. 4. Fitness and growth for K=2 and various C with larger $N0$ and various g

as used above and $p_g = p_c$ as before. It can be seen that there is no significant difference in performance but that an increase in the rate of growth is experienced. The growth appears to be driven by slight (sustainable) increases in the frequency parameters, this being more rapid the higher C. This was found to be the case for all K tried (not shown). Therefore the system appears able to exploit the findings of the previous section where it was shown that the size or amount of sustainable growth is directly proportional to the original genome length. That is, increased growth is experienced due to the dynamic adjustment of the frequency as the average genome length increases. Results with higher initial seeds, e.g., [0,1], find, as predicted by results above, that the typical frequency of change is too great to enable sustainable growth (not shown).

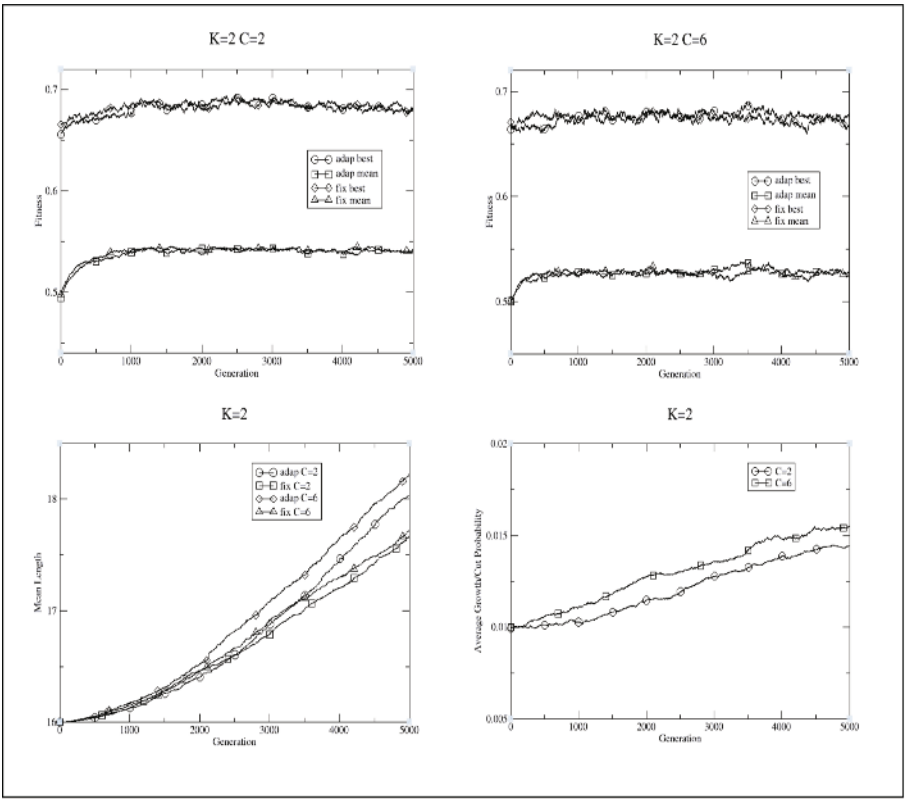


Fig. 5. Behaviour for K=2 and various C with self-adaptive growth

7 Conclusions

The Species Adaptation Genetic Algorithm was introduced to enable open-ended evolution within artificial systems. Recently, this idea has been applied to coevolutionary systems. In this paper it has been shown that the dynamics of the coupled fitness landscapes of coevolutionary systems can enable more sustainable growth within a SAGA the higher the degree of coupling. The amount of sustainable growth is directly proportional to the length of the original genome. Further, the use of

a self-adaptive growth mechanism confirms this general behaviour for such systems and enables more rapid growth. Here the rate of growth increases with increasing genome length; growth begets more growth.

References

1. Axlerod, R. (1984) *The Evolution of Cooperation*. Basic Books.
2. Bull, L. (1997) On the Evolution of Multicellularity. In P. Husbands & I. Harvey (Eds) *Proceedings of the Fourth European Conference on Artificial Life*. MIT Press, pp190-196.
3. Bull, L. (1997) Evolutionary Computing in Multi-Agent Environments: Partners. In T.Baeck (Ed) *Proceedings of the Seventh International Conference on Genetic Algorithms*. Morgan Kaufmann, pp370-377.
4. Bull, L. (1999) On the Evolution of Multicellularity and Eusociality. *Artificial Life* 5(1):1-15.
5. Bull, L. & Fogarty, T.C. (1996) Artificial Symbiogenesis. *Artificial Life* 2(3):269-292.
6. Bull, L., Fogarty, T.C. & Pipe, A.G. (1995) Artificial Endosymbiosis. In F. Moran, A. Mereno, J.J. Merelo & P. Chacon (Eds) *Advances in Artificial Life - Proceedings of the Third European Conference on Artificial Life*. Springer-Verlag, pp273-289.
7. Bull, L., Holland, O. & Blackmore, S. (2000) On Meme-Gene Coevolution. *Artificial Life* 6(3): 227-235.
8. Harvey, I. (1992) Species Adaptation Genetic Algorithms: A Basis for a Continuing SAGA. In F.J. Varela & P. Bourgine (Eds) *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*. MIT Press, pp346-354.
9. Harvey, I., Cliff, D.&Husbands, P. (1994) See ing the Light: Artificial Evolution, Real Vision. In D. Cliff, P. Husbands, J-A. Meyer & S.W. Wilson (Eds) *From Animals to Animats 3*. MIT Press, pp392-401.
10. Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press.
11. Holland, J.H. (1976) Adaptation. In R. Rosen & F.M. Snell (Eds) *Progress in Theoretical Biology*. Plenum.
12. Hurst, J. & Bull, L. (2005) A Neural Learning Classifier System with Self-Adaptive Constructivism for Mobile Robot Control. *Artificial Life* (in press).
13. Kauffman, S.A. & Levin, S. (1987) Towards a General Theory of Adaptive Walks on Rugged Landscapes. *J. Theoretical Biology* 128: 11-45.
14. Kauffman, S.A. & Johnsen, S. (1991) Coevolution to the Edge of Chaos: Coupled Fitness Landscapes, Poised States and Coevolutionary Avalanches. In C.G. Langton, C. Taylor, J.D. Farmer & S. Rasmussen (Eds) *Artificial Life II*. Addison-Wesley, pp325-370.
15. Khakhina, L.N. (1992) (Ed) *Concepts of Symbiogensis: History of Symbiogenesis as an Evolutionary Mechanism*. Yale University Press.
16. Koza, J.R. (1994) *Genetic Programming*. MIT Press.
17. Lindgren, K. (1991) Evolutionary Phenomena in Simple Dynamics. In C.G. Langton, C. Taylor, J.D. Farmer & S. Rasmussen (Eds) *Artificial Life II*. Addison-Wesley, pp.295-312.
18. Maynard Smith, J. & Szathmary, E. (1995) *The Major Transitions in Evolution*, W H Freeman.
19. Quinn, M., Smith, L., Mayley, G. & Husbands, P. (2002) Evolving Teamwork and Role Allocation for Real Robots. In R.K. Standish, M.A. Bedau & H.A.Abbass (Eds) *Proceedings of the Eighth International Conference on Artificial Life*. MIT Press, pp302-311.
20. Smith, S.F. (1980) A Learning System Based on Genetic Adaptive Algorithms. PhD Thesis, University of Pittsburgh.
21. Smith, J. & Smith, R.E. (1999) An Examination of Tuneable Random Search Landscapes. In W. Banzhaf & C. Reeves (Eds) *Foundations of Genetic Algorithms V*. Morgan Kauffman, pp165-182.
22. Tomlinson, A. & Bull, L. (2001) Symbiogenesis in Learning Classifier Systems. *Artificial Life* 7(1):33-62.

Evolution and the Regulation of Environmental Variables

Hywel Williams and Jason Noble

School of Computing, University of Leeds, Leeds UK
{hywelw, jasonn}@comp.leeds.ac.uk

Abstract. The idea that the biota can regulate the abiotic components of their environment to levels suitable for life has attracted criticism from neo-Darwinian theorists but is still a viable hypothesis. Here we present a model, similar to Daisyworld [1] but more general, which allows for a more extensive study of the compatibility of biotic regulation with evolutionary theory. Results obtained highlight the importance of constraints on the evolutionary process for the emergence of regulation, and set the scene for more comprehensive future study.

1 Introduction

The Gaia hypothesis, initially proposed by Lovelock and Margulis in 1973 [2] and developed further by subsequent debate in the literature, postulates the regulation of the abiotic environment by the biota so that the biosphere is maintained in conditions suitable for life. Although the initial idea of biotic feedback concerned the chemical composition of the atmosphere, the concept has been extended to other features of the Earth system, such as the carbon and nitrogen cycles or the N:P ratio in the oceans [3]. A significant amount of empirical evidence has been gathered on various facets of the Earth system, but the scale of the system under study and the sample size (of one!) has meant that no conclusive evidence has been found to either prove or disprove the hypothesis. Debate of the Gaia theory has therefore been largely theoretical, concerning the plausibility of such planet-wide regulatory mechanisms and the compatibility of regulation with Darwinian evolution.

The neo-Darwinian critique has shifted focus over the years. Early criticisms of the Gaia theory claiming teleology were rebuffed by the Daisyworld model [1], a numerical thought experiment which demonstrated how the temperature of a simulated planet might be regulated in the face of increasing solar radiation by a process of ecological competition between black and white daisy species. Since then the Daisyworld model has become the focus of debate; lacking a more approachable target in the real Earth system, protagonists in the theoretical debate have been forced to fight many of their battles over a model that was originally intended as a simple proof of concept.

While it should always be remembered that a Daisyworld is not a real world, a number of key conceptual clarifications have been made by its use. However,

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 332–341, 2005.

the simplicity of the original Daisyworld model means that it cannot answer questions about the compatibility of biotic regulation and evolution. To address these questions a number of extensions to the model have been presented [4,5,6,7,8,9,10], but it seems likely that the Daisyworld model has now answered as many questions as it is able to.

Biotic regulation of abiotic variables is a general and widely applicable concept (see also the extensive literature concerning the related concepts of niche construction [11] and the 'extended phenotype' [12]), that extends far beyond daisies and temperature regulation, but it is not yet fully understood. Nowadays not many scientists would dispute that life affects the physical environment (itself a radical claim when Lovelock and Margulis first stirred up the Gaia debate), and vice versa, but the nature of the feedback between the two is unclear (witness the ongoing Gaian debate in the climate change literature [13]). It is difficult to study such processes in the real world, because of the aforementioned problems of scale and sample size, but simulation modelling can offer a useful tool. While models cannot prove anything about the real world, they can focus debate, generate hypotheses and allow the testing of assumptions as 'opaque thought experiments' [14].

This paper presents the first step in a longer-term project to model and explore how biotic regulation of abiotic environmental variables may evolve. The bulk of the paper will be concerned with developing a simplified version of Daisyworld that captures all known results. We further simplify the 'cut-down Daisyworld' model presented by Harvey [15] and extend it to a 2-dimensional cellular automata model amenable to the inclusion of Darwinian evolution. Existing results from earlier Daisyworld models are reiterated before presentation of some new results concerning the necessity of constraints on evolution if regulation is to emerge.

2 The Model

2.1 Overview

The original Daisyworld model [1] incorporates two species of daisy, identical except that one is black (with low albedo) and the other is white (with high albedo). Daisy albedo alters the local temperature of each daisy patch, with daisies assumed to live in single-species clumps large enough to maintain their own local temperature. This in turn alters the growth rate of the daisies, which varies as a function of temperature. Since the albedo of bare earth lies between the albedos of black and white daisies, population dynamics allow global temperature to move away from that expected of a dead planet. Competition between black and white daisies led to global temperature regulation around the optimal temperature for daisy growth; deviations away from this point were counteracted by negative feedback engendered by the selective advantage gained by one of the daisy species away from this point. Black daisies out-compete white daisies at low temperatures because of their ability to increase local temperature, and vice versa at high temperatures. Regulation was observed for a significant range of

solar luminosity, outside which the planet was too cold or too hot to support daisies of any colour.

The Daisyworld model used reasonably accurate approximations of the real-world phenomena on which it was based and thus incorporated quite complicated mathematical formulations of (for example) the interaction between solar luminosity and the level of heat radiation emitted by the planet. Harvey [15] developed a simplified model, his 'cut-down Daisyworld', that used much simpler approximations but conserved the essential regulatory behaviour of the system. Following Harvey and simplifying even further, we present the very basic model described below.

Our model is a cellular automata model in which patches are arranged in a 2-dimensional toroidal lattice (another CA Daisyworld model was presented by von Bloh et al [6], but in a different form and with different aims). Each patch may be barren (bare earth) or may contain a single species of daisy. Barren patches can be colonised by daisies from neighbouring patches, while living patches may die. Each patch has a local temperature that changes in relation to solar luminosity (applied at an equal level to all patches) and to its albedo (determined by the presence of daisies). The global temperature of the planet is taken as the mean of all the local patch temperatures. This scheme is covered in more detail below.

2.2 Daisies

A daisy species is represented by an albedo and a growth function. Albedo is drawn from the range $[0.25, 0.75]$ (representing a continuum from black to white), while the growth function is a piecewise linear function of local temperature that has the qualitative form shown below in figure 1. This hat-shaped function can be represented by the location of its centre point H_{mid}. In all the experiments reported here the hat function reached zero at $H_{mid} \pm 15$.

Daisies do not grow as such, since they are assumed to either fully occupy a patch or not to be present, but the growth rate of a daisy species determines its likelihood of colonising a neighbouring bare patch. High growth rates lead to increased colonisation.

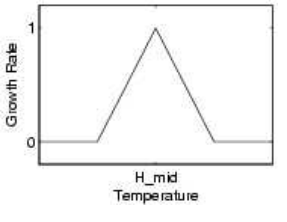


Fig. 1. An example growth function. The growth rate of all daisy species varies from 0 to 1 as a piecewise linear function of temperature.

2.3 Seeding

An empty patch may be seeded with a new daisy species with low probability (0.03 in the simulations described below). When seeding occurs an entirely new daisy type is randomly generated from the set of permissible values for albedo and growth function parameters. Seeding allows new genetic stock to gain a foothold in the world and takes the place of mutation in the evolutionary process.

2.4 Colonisation

Empty patches may be colonised by daisy species living in neighbouring patches. Each neighbour species has a chance to colonise that is proportionate to its growth rate. This is implemented by assigning a probability $P(C_i)$ to the event that the empty patch is colonised by the i^{th} neighbouring patch (alive or dead), such that $P(C_i) = \frac{G_i}{N}$ (where G_i is the growth rate of the i^{th} neighbouring patch and N is the total number of neighbours) and noting that the growth rate of a dead patch is zero. Thus daisies with a higher growth rate have a higher likelihood of colonisation. Also, a daisy species occupying multiple neighbouring patches has a higher likelihood of colonisation due to having more 'tickets in the lottery'.

2.5 Death

If a daisy species living in a patch has a growth rate of zero, it is assumed not to be able to survive and the patch becomes empty. Also, daisies living in a patch will die (and the patch become empty) with a probability of 0.1 at each timestep. This may be seen as a simple instantiation of death by natural causes and serves to promote selection and competition.

2.6 Calculation of Patch Temperature

Local patch temperature depends on the current temperature of the Sun (traditionally taken in Daisyworld models as a monotonically increasing value), the albedo of the patch (determined by daisy growth), and heat loss to space. The rate of change of local patch temperature is therefore given by equation 1 below, where T_P is the patch temperature, T_S is the temperature of the Sun, and α is the patch albedo.

$$\frac{dT_P}{dt} = (1 - \alpha)(T_S - T_P) - T_P \tag{1}$$

Patch temperatures are integrated numerically using Euler's forward method. The global temperature of the planet is taken as the mean of all the patch temperatures.

2.7 Cellular Automata Update

The results presented below were gathered from a 10x10 toroidal CA where each patch has 4 neighbours at top, bottom, left and right. The CA is synchronously updated at each timestep by testing for colonisation, seeding and death in that order. T_S is typically increased from 100 to 500 in increments of 2, and the CA is updated for 1000 timesteps for each increment in T_S to allow the daisy population to stabilise for the new level of external forcing.

3 Repetition of Existing Daisyworld Results

First of all we compared the results generated from our model with known results generated from existing Daisyworld model. In all of the following experiments the albedo of bare earth was set to 0.5 and the world was initialised with all patches bare.

The primary Daisyworld phenomenon, that of temperature regulation by competition between black and white daisies [1], was considered first of all. We set the albedo of black daisies to 0.25 and the albedo of white daisies to 0.75. Results are shown in figure 2, which displays global temperature regulation occurring by competition between the daisy species as it does in the original work.

The next significant result to be repeated is that allowing albedo to mutate does not affect regulation, and may in some cases actually increase its range [4,5,6,7]. For this scenario we allowed albedo to take any value in the range $[0.25, 0.75]$, corresponding to the full range from black to white. Temperature was regulated as before, although in this case it is by a steady shift in the albedo of the dominant daisy species to maintain the global temperature close to the optimal level, rather than competition between black and white daisies. The overall effect is the same at a global level; temperature regulation in this case and in the previous case is achieved by keeping the mean global albedo close to the level which keeps temperature optimal. This in turn is a result of selection for the daisy species with the highest growth rates.

Having shown that temperature regulation is not affected by mutation of albedo, the next result is to show that unconstrained mutation of the growth

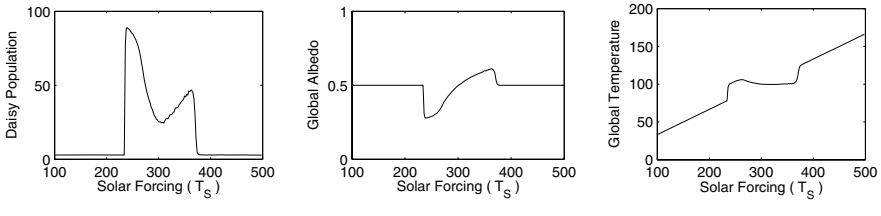


Fig. 2. Daisy population, global albedo and global temperature for a world with both black (albedo = 0.25) and white (albedo = 0.75) daisies

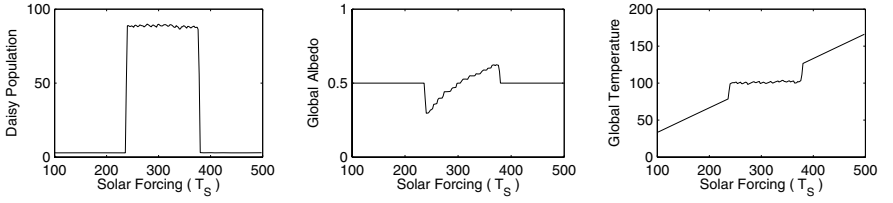


Fig. 3. Daisy population, global albedo and global temperature for a world where daisy albedo is allowed to mutate freely between the levels for black and white daisies, i.e., within the range $[0.25, 0.75]$

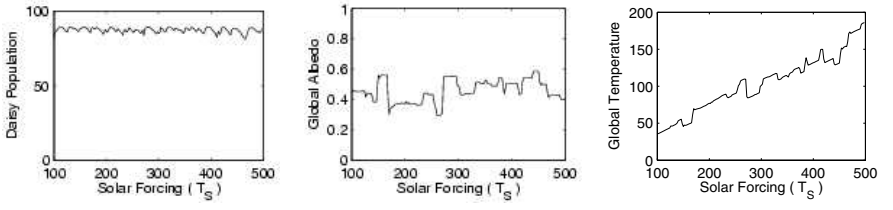


Fig. 4. Daisy population, global albedo and global temperature for a world where daisy albedo is allowed to mutate freely between the levels for black and white daisies, i.e., within the range $[0.25, 0.75]$ and where the centre of the growth function is allowed to mutate freely

function causes the regulation to break down [9]. Here we do this by allowing H_{mid} to vary freely in the range $[70, 130]$. No regulation is observed, although the daisy population flourishes. The mutation in growth function simply tracks the solar forcing; the daisies adapt themselves to the environment rather than adapt the environment to themselves. The tracking is not precise, as the mutating albedo allows brief periods of quasi-regulation when the population becomes fixated on a particular growth function and uses the albedo to maintain the global temperature this value of H_{mid} requires. In this scenario albedo can be seen as a free variable, since a suitable growth function can be found to give optimal growth for any albedo level.

Lenton and Lovelock [10] showed that when there is some constraint on the mutation of the growth function, regulation will again emerge. They set up a Daisyworld model where the centre of the hat function was mutated towards the current ambient temperature, but where the maximum growth rate (i.e., the maximum height of the growth function) fell away to zero with distance from some optimal value, in a way supposed to be analogous with the decline in maximum achievable photosynthesis rate varies with temperature in plants. We implemented this by letting the maximum growth rate decline linearly to zero with distance from an optimal temperature of 100. We observed similar results to Lenton and Lovelock [10], in that regulation was observed to occur, but with a more gradual tailing in and tailing out than with the non-evolvable growth function.

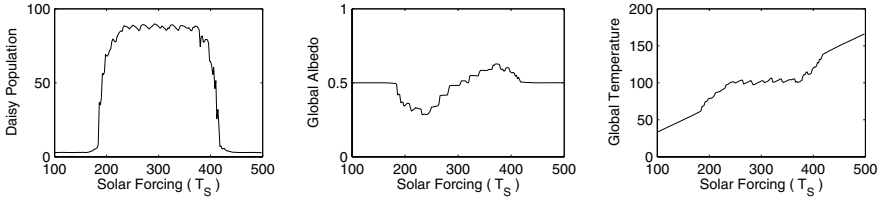


Fig. 5. Daisy population, global albedo and global temperature for a world where daisy albedo is allowed to mutate freely between the levels for black and white daisies, i.e., within the range $[0.25, 0.75]$ and where the centre of the growth function is allowed to mutate freely. Maximum achievable growth rate declines linearly with distance from $T = 100$, reaching zero at $T = 100 \pm 30$.

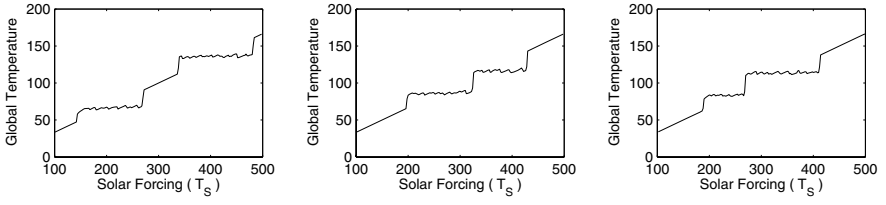


Fig. 6. Regulation of global temperature for a world where there are two permissible growth functions and where daisy albedo is allowed to mutate freely between the levels for black and white daisies, i.e., within the range $[0.25, 0.75]$. Plots are shown for well-separated growth functions (centres at T=65 and T=135) 6(a) and for growth functions with overlapping ranges (centres at T=85 and T=115) where solar forcing increases 6(b) and decreases 6(c) over time.

For completeness, we have also run the model with heat transfer between neighbouring patches [6], and found that the qualitative nature of the results is unchanged for all of the above scenarios. The exception is where the heat transfer is so efficient that there is no perceptible difference in temperature between patches; in this case there is no possiblity of a particular daisy species gaining a selective advantage over its competitors by altering its local temperature. When this occurs the regulation of global temperature is lost.

4 Constraints on Evolution and Their Implications for Environmental Regulation

It seems that the key criteria for regulation of global temperature to emerge are *need* and *ability*. Unless there is some reason for the daisies to alter their local environment, i.e., some selective advantage to be gained from doing so, then regulation will not occur. If evolution is added to the model, then the only cases in which daisies have a reason to alter their environment are those in which the evolutionary process is constrained in some way so that the daisy population cannot evolve to prefer the environment as it is. Selective advantage is gained

by improving the fit between daisy and environment; this can be achieved by changing the daisy or by changing the environment, and evolution will generally opt for the easiest method available.

Constraints on evolution are an inevitable feature of any real-world biological system, due to the existence of physical and chemical laws that no system may violate. Chemical laws constrain metabolism, the rate of which typically depends on a number of parameters as some bell-shaped curve. This idea is captured simply in Daisyworld as a growth function that depends on temperature by a Gaussian function, and in the current model by a piecewise linear hat function. While evolution cannot alter the chemical reactions involved in metabolism, it may tinker with the conditions under which those reactions operate to maximise their rate and efficiency, or it may select between different sets of reactions, that is, between different types of metabolism. However, once a metabolism has been chosen during the course of evolution it may often be easier to regulate the environment to suit this metabolism than to switch to a new metabolism entirely.

Different metabolic types may be more successful at different ranges of an environmental variable. In our simplified Daisyworld model, consider a situation where there are two growth functions with centres at different temperatures. The different growth functions may be well-separated, leading to independent regulatory epochs (figure 6), or have overlapping ranges, leading to competitive exclusion (figures 6(b) and 6(c)). When ranges overlap there will usually be one dominant metabolic type around which the environment is regulated, with a flip from one to the other at some critical level of solar forcing. The level at which this occurs depends on the history of the system. Whichever metabolic type becomes abundant first will stop the late-comer from getting a foot-hold in the ecology by holding temperature close to its own optimal level, and thus delay the onset of an ecology (and regulation) based around the other type. This is demonstrated by figures 6(b) and 6(c) which show competition between two growth functions with overlapping ranges in the face of increasing and decreasing solar forcing respectively (i.e., time flows to the right in 6(b) and to the left in 6(c), although forcing is plotted increasing left-to-right in both).

Another way in which evolution may be constrained and create an opportunity for regulation to evolve is if evolution operates at different rates on different phenotypic traits. Consider the case where the daisy growth function is free to mutate so that it can operate at any temperature and where daisy albedo may also mutate freely to any level between those for black and for white daisies. If both types of mutation occur at the same rate, then the growth function simply tracks the increasing solar forcing and regulation is lost (figure 4). However, if the mutation rate for the growth function is very slow compared to mutation rate of albedo the differential creates an opportunity for regulation. It is easier for a daisy species to evolve a new albedo than a new growth function. This can be observed in figure 7, in which the world is started with a viable daisy population that is then allowed to mutate. At each daisy reproduction (each colonisation of an empty patch), the daisy species may mutate its growth

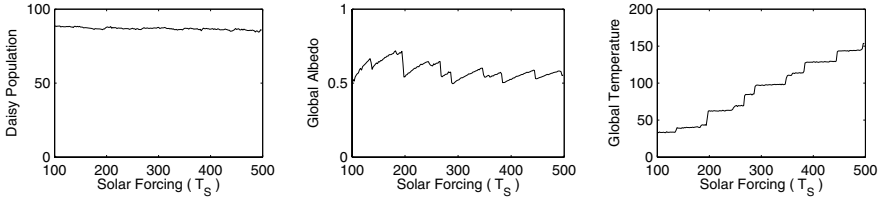


Fig. 7. Daisy population, global albedo and global temperature for a world where daisy albedo is allowed to mutate freely between the levels for black and white daisies (i.e., within the range $[0.25, 0.75)$ with a probability of 0.2 at each reproduction and where daisy growth function can mutate freely with a probability of 0.002 at each reproduction.

function with probability 0.002 and its albedo with probability 0.2 (so albedo mutates two orders of magnitude faster than the growth function). As can be seen from figure 7, this results in regulatory epochs where the daisy population regulates the global temperature around the optimum for some growth function. Eventually the albedo can mutate no further and mutants with a more suitable growth function can out-compete the existing population to become established as the new dominant metabolic type around which regulation occurs.

5 Conclusion

We have presented a model that is derived from Daisyworld, but is simplified and extended to allow for a more comprehensive study of the compatability of biotic environmental regulation with evolutionary theory. The model has been described here using the language of Daisyworld (daisies using albedo to regulate temperature in the face of solar forcing), but the mathematical formulation of the model is actually very general, allowing its possible use to study the concept of biotic regulation as a general phenomenon. Our model shows that what is needed for regulation to emerge are constraints on the evolutionary process and the possibility of organisms creating some local buffer against the global environment, criteria that we feel are plausible in a wide variety of biological systems. In future work we hope to move away from the daisy metaphor and look at multi-dimensional regulation in a more general sense.

References

1. Watson, A., Lovelock, J.: Biological homeostasis of the global environment: the parable of daisyworld. Tellus **35B** (1983) 284–289
2. Lovelock, J., Margulis, L.: Atmospheric homeostasis by and for the biosphere: the gaia hypothesis. Tellus **26(2)** (1973) 2–9
3. Volk, T.: Gaia's Body: Toward a Physiology of Earth. Springer-Verlag, New York (1998)
4. Lovelock, J.: A numerical model for biodiversity. Philosophical Transactions of the Royal Society of London: Series B **338** (1992) 383–391

5. Stocker, S.: Regarding mutations in daisyworld models. Journal of Theoretical Biology **175** (1995) 495–501
6. von Bloh, W., Block, A., Schellnhuber, H.: Self-stabilization of the biosphere under global change: a tutorial geophysiological approach. Tellus **49B** (1997) 249–262
7. Lenton, T.: Gaia and natural selection. Nature **394** (1998) 439–447
8. Saunders, P.: Evolution without natural selection: further implications of the daisyworld parable. Journal of Theoretical Biology **166** (1994) 365–373
9. Robertson, D., Robinson, J.: Darwinian daisyworld. Journal of Theoretical Biology **195** (1998) 129–134
10. Lenton, T., Lovelock, J.E.: Daisyworld is darwinian: constraints on adaptation are important for planetary self-regulation. Journal of Theoretical Biology **206** (2000) 109–114
11. Laland, K., Odling-Smee, F., Feldman, M.: Niche construction, biological evolution and cultural change. Behavioral and Brain Sciences **23** (1999) 131175
12. Dawkins, R.: The Extended Phenotype. Oxford University Press, Oxford (1982)
13. Volk, T.: Toward a future for gaia theory. Climatic Change **52** (2002) 423–430
14. Di Paolo, E., Noble, J., Bullock, S.: Simulation models as opaque thought experiments. In Bedau, M., McCaskill, J., Packard, N., Rasmussen, S., eds.: Artificial Life VII: The Seventh International Conference on the Simulation and Synthesis of Living Systems, Cambridge, MA, MIT Press/Bradford Books (2000) 497–506
15. Harvey, I.: Homeostasis and rein control: From daisyworld to active perception. In Pollack, J., Bedau, M., Husbands, P., Ikegami, T., Watson, R., eds.: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, ALIFE'9, Cambridge, MA, MIT Press (2004) 309–314

Evolutionary Transitions as a Metaphor for Evolutionary Optimisation

Anne Defaweux[1], Tom Lenaerts[2], and Jano van Hemert[3]

[1] COMO, Vrije Universiteit Brussel, Brussels, Belgium
[2] IRIDIA, CP 194/6, Université Libre de Bruxelles, Brussels, Belgium
[3] Center for Emergent Computing, Napier University, Edinburgh, United Kingdom
adefaweu@vub.ac.be, tlenaert@ulb.ac.be

Abstract. This paper proposes a computational model for solving optimisation problems that mimics the principle of evolutionary transitions in individual complexity. More specifically it incorporates mechanisms for the emergence of increasingly complex individuals from the interaction of more simple ones. The biological principles for transition are outlined and mapped onto an evolutionary computation context. The class of binary constraint satisfaction problems is used to illustrate the transition mechanism.

1 Introduction

From biological literature one can learn that life is organised in a hierarchical fashion and that transitions in complexity have occurred linking the different levels of this hierarchy. Typical examples in this context are the transitions from genes to simple cells, from single cells to multi-cellular organisms or from single organisms to social systems [1,2]. It has been argued that these transitions in the complexity of the evolving individuals share two common themes: (1) the emergence of cooperation among individuals at a lower level in the hierarchy into the functioning of a new higher level unit and (2) the regulation of conflict among these lower level units.

In this article, the metaphor defined by transitions in biological complexity is used to construct an artificial evolutionary system which can be used in the context of optimisation and learning. The central problem we investigate is how a system can be designed that captures the two themes of cooperation and mediation proposed by Michod [2] into a suitable algorithm. Hence, this article will discuss a mapping between the abstract scheme that captures the common structure of evolutionary transitions and an artificial evolutionary model that can serve as an alternative for the simple genetic algorithm (GA).

We focus here on system that provides transitions in the context of the solution complexity for the class of binary constraint satisfaction problems (BINCSP). This example from optimisation was chosen for four reasons: (1) we were interested in a problem where solutions can be modelled by the aggregation of lower level (partial) solutions, (2) cooperative interactions between the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 342–352, 2005.

partial solutions can be defined in a natural way, (3) when the interactions benefit the partners, the new unit of selection that emerges at the higher-level can still be interpreted using semantics defined by the problem under observation, and (4) previous studies enable us to create problems with a controlled level of difficulty [3]. Consequently from the first three reasons, the emergent unit still has some meaningful functionality in the context of the problem.

The difference with the GA approach to evolution is that solutions are variable length representations which increase in complexity, individuals use only replication and mutation and are placed in an interactive framework which supports collaborative behaviour. Consequently, the proposed model is related to messy Genetic Algorithms (mGA) [4] and the Compositional Evolution model [5,6]. For details on the technical differences, we refer to [7]. Conceptually, the difference is in the metaphor used to construct the model. Here, as mentioned earlier, the transition perspective focuses on the one defined in [2].

In the next section, the class of optimisation problems for which the transition model will be defined is explained. Given this problem context, a mapping is examined between the transition cycle and the proposed evolutionary optimisation system. Afterwards an illustrative experiment is performed to demonstrate the increase of complexity and its effect on the fitness.

2 Optimisation Context for Transition Study

Constraint Satisfaction Problems (CSP) [8] form a NP-complete problem class where, on the one hand, one has a set of variables X associated with possible domain values D and, on the other hand, a set of constraints C defined on this set of variables, which prohibits combinations of assignments to occur. The problem consists in finding an assignment to the whole set of variables from the associated domain values so that all constraints are satisfied. If this proves to be impossible then the corresponding problem is said to be unsolvable.

A variant of this problem is BINCSP, where each constraint is defined on at most two variables. This forms no restriction on the general form of CSP as every CSP can be rewritten into a BINCSP and vice versa [9].

Let us take as an illustration the following BINCSP: consider a set of six variables: $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ all taking values in $D = \{1, 2, 3\}$. We consider the following set of constraints:

$$\begin{aligned} C = \{&(x_1 \neq x_2), (x_2 \neq x_3), (x_3 \neq x_1), \\ &(x_4 \neq x_5), (x_5 \neq x_6), (x_6 \neq x_4), \\ &(x_1 = x_4), (x_2 = x_5), (x_3 = x_6)\} \end{aligned} \tag{1}$$

This setup of constraints consists of nine binary constraints. Each binary constraint defines a relation between two variables of X. Also, for each pair of variables, only one binary constraint may be defined.

The problem involves finding the correct assignment for the variables so that all these constraints are satisfied. We denote the assignment of one variable $x_i \in X$ with value $d \in D$ by $\langle d, i \rangle$ where i is the index of the variable we consider.

Using this notation, we represent the simultaneous assignment of variables x_1, x_2 and x_4 with respective values v_1, v_2 and v_4 as

$$(\langle v_1, 1\rangle, \langle v_2, 2\rangle, \langle v_4, 4\rangle) \tag{2}$$

A *solution* for a BINCSP problem consists in an assignment of variables from X to values of D. We use randomly generated problem instances of BINCSP. The RandomCSP package [10] is used to generate the suite of test problem instances [11]. To scale the difficulty of the problem instances, these CSP are generated according to two parameters. For more details see [11,7]. An important property that was observed is that for certain problem instances there is more structure in the search space than others. Whenever structure is present the algorithm described later can exploit it.

3 Evolutionary Transition in BINCSP Solutions

As mentioned in the introduction, all transitions in nature share two common themes: cooperation and conflict mediation among the lower-level individuals. These themes are captured in the transition cycle visualised in Figure 1. One can observe four phases in this cycle and these phases need to be captured by the proposed algorithm.

A system that uses the metaphor visualised in the figure, should be able to apply it iteratively. In other words, repeated phases of cooperation and mediation between ever increasing levels will produce more and more complex organisms which try to survive in their selective environment. In the following sections a mapping will be defined of an optimisation process onto the cycle. For the rest of the discussion, it is assumed that an evolutionary system is present that simulates the process of differential survival and reproduction of the partial and complete solutions for a particular BINCSP problem.

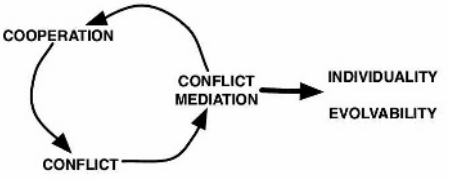


Fig. 1. Transition Cycle [2]; Every transition starts at a certain level of complexity. At this level cooperation needs to emerge since it exchanges fitness at the lower level with fitness at the higher level. Yet conflict remains. Defection among the lower-level units can lead to the destruction of the cooperative group. These conflicts need to be mediated and this will lead to a new level of individuality with its own heritable variations that evolve and diversify.

3.1 Representation of Lower-Level Units

At the lowest (initial) level, the system consists of a population P containing N individuals where each individual i is represented by a set of variables $S_i \subseteq X$ for all $i \in \{1, 2, ..., N\}$. Here it is assumed that, at the lowest level, the size of this set S is 2. Hence, the initial population contains only *partial* solutions which can solve one of the binary constraints in the set C (see Equation (1)). An individual which contains a value for all variables in X is referred to as a *complete* or *fully qualifying* solution. Hence, *complexity* in the current system refers to the number of variables present in an individual i.e. individuals of maximum complexity are complete solutions.

A partial solution s that only defines values for x_1 and x_2 is for example:

$$(\langle 1, 1\rangle, \langle 3, 2\rangle). \tag{3}$$

(3) is called the *genotype* of the solution. The selective system will operate on the quality of the genotype in solving the constraints listed in the set C.

3.2 Cooperation Between Lower-Level Units

Interactions between the partial individuals is done between pairs of individuals that are randomly selected from the population P. The experiments discussed here will not consider larger groups. This assumption is removed in some ongoing experiments, yet they will not be reported here.

Interaction between partial solutions is defined in the following manner. Let solution s defined by (3) interact with a symbiotic partner sp defined by $(\langle 3, 1\rangle, \langle 2, 3\rangle)$. This interacting partner is referred to as the symbiotic partner of (3) and we denote the relation by:

$$(\langle 1, 1\rangle, \langle 3, 2\rangle) \leftrightarrow (\langle 3, 1\rangle, \langle 2, 3\rangle). \tag{4}$$

We simulate interaction between 2 partial solutions by sharing the information contained in their genotypes. The outcome of the information sharing between a solution and its symbiotic partner is called the *phenotype* of the solution. For example, the phenotype of (4) is simply obtained by combining all information present in both genotypes:

$$\langle \begin{pmatrix} 1 \\ 3 \end{pmatrix}, 1\rangle, \langle 3, 2\rangle, \langle 2, 3\rangle \tag{5}$$

Important to notice here is that the genetic information of both individuals is not changed. The heritable capacity of both s and sp remains at the level of the simple units.

The situations one can have when combining s and sp correspond to the different general forms of symbiosis: parasitism (P) , mutualism (M) , commensalism (C) and amensalism (A). In the case of parasitism, as shown in Table 1, the association is disadvantageous for one of the partners and beneficial to the

other one. The outcome of the interaction between s and sp is parasitic if s solves one of the constraints of C correctly ($A(s)$ =high) and sp does not ($A(sp)$ =low). The information sharing will in that case benefit sp since it increases its adaptiveness and it decreases the adaptiveness of s. The relation between s and sp is mutualistic if both partners gain something from the relation ($A(s, sp)$ =high). In the table, both individuals have low adaptiveness but when the two variables x_2 and x_3 are combined, i.e. ($\langle 3, 2\rangle, \langle 2, 3\rangle$), their adaptiveness increases. Commensalism occurs when the adaptiveness of one of the partners does not change due to the information sharing. In the table, an example is shown where both individuals have a value for variable x_1 i.e. the values 1 and 3. Now if the assignment $x_1 = 1$ resolves one of the constraints and it is assumed that the value is selected by both partners then only sp benefits from the relation and things do not change for s. A similar reasoning can be followed for amensalism. In that case the association is disadvantageous for one of the partners. Yet then instead of choosing the best value for x_1, the worst one is selected.

3.3 Conflict Mediation

Although cooperative behaviour produces better results in the long term, short term considerations will lead to defecting behaviour. An important choice made by individuals in a transition model is whether they will share the information or not. In order to have transitions in complexity, mechanisms should be put into place which encourage the evolution of information sharing behaviour. In the current model, it assumed that individuals want to collaborate. In other words they are all cooperative. In further experiments this assumption is relaxed. This simplification was made to examine whether cooperative partially defined units can actually lead to fully qualifying solutions for BINCSP problems. In general, principles from multilevel selection are incorporated to model the evolution of cooperative interactions between pairs (or between members of more complex groups) [12]. For now, we focus on another conflict issue.

Next to the choice of collaborating or not, other conflicts can occur. As shown in (5) partners can have different values for the same variables. These

Table 1. Some examples of the different forms of symbiosis and their relation to the BINCSP problem. $A(s)$ and $A(sp)$ evaluate the adaptiveness of both individuals in their personal relation to the problem.$A(s, sp)$ refers the effects of the information sharing on the adaptiveness of both individuals.

	s	$A(s)$	sp	$A(sp)$	symbiosis	$A(s, sp)$
P	$(\langle 1,1\rangle, \langle 3,2\rangle)$	high	$(\langle 3,3\rangle, \langle 2,4\rangle)$	low	$(\langle 1,1\rangle, \langle 3,2\rangle, \langle 3,3\rangle, \langle 2,4\rangle)$	(low,high)
M	$(\langle 1,1\rangle, \langle 3,2\rangle)$	low	$(\langle 3,1\rangle, \langle 2,3\rangle)$	low	$(\langle 1,1\rangle, \langle 3,2\rangle, \langle 3,1\rangle, \langle 2,3\rangle)$	(high,high)
C	$(\langle 1,1\rangle, \langle 3,2\rangle)$	high	$(\langle 3,1\rangle, \langle 2,3\rangle)$	low	$\langle \begin{pmatrix} 1 \\ 3 \end{pmatrix}, 1\rangle, \langle 3,2\rangle, \langle 2,3\rangle$	(high,high)
A	$(\langle 1,1\rangle, \langle 3,2\rangle)$	high	$(\langle 3,1\rangle, \langle 2,3\rangle)$	low	$\langle \begin{pmatrix} 1 \\ 3 \end{pmatrix}, 1\rangle, \langle 3,2\rangle, \langle 2,3\rangle$	(low,low)

problems with conflicting values are resolved by selecting randomly one of the possible choices. Hence, the symbiotic behaviour can correspond to any of those described in Table 1.

In our example, a conflict needs to be resolved for variable x_1. We can choose between the values 1 and 3. A possible conflict resolution in this case would be:

$$\langle 1, 1\rangle, \langle 3, 2\rangle, \langle 2, 3\rangle \tag{6}$$

(6) is called the *induced phenotype* of the partial solution (3). This phenotype is used for evaluation and the result of the evaluation is assigned to the genotype s i.e. (3). We denote the phenotype of a solution s interacting with sp by: $\phi(s, sp)$.

The phenotype assigned to the symbiotic partner sp is obtained in the same way. Yet, the policy about conflicting values may yield another representation than the one we obtained for the initial partial solution s. This asymmetry between the phenotype of a solution and the phenotype of its symbiotic partner increases the exploration possibilities of the evolutionary process. Note that the conflict mediation strategy adopted for this particular test case avoids the system to build greater genotype than the maximum size expected for a genotype. Hence, the problem related to ever growing genotypes which is a classical issue in variable length representation does not occur here.

3.4 Intermezzo: Evaluation of Genotypes

Here two types of functions are considered. One function to determine the success of the solution in terms of the complete constraint set ($f(s)$) and another function to determine how good it scores relative to the constraints it covers ($f_{cov}(s)$).

Assume that $c_k(p)$ is the outcome of evaluating phenotype p with constraints k, we say that p covers c_k if p contains assignments for all variables contained in c_k, furthermore, p satisfies c_k if the assignment values in p do not violate the constraints defined by c_k.

$$c_k(p) = \begin{cases} 1 \text{ if } p \text{ covers } c_k \ \wedge p \text{ satisfies } c_k \\ 0 \text{ otherwise} \end{cases} \tag{7}$$

Given this, the classical evaluation of the solution described by (3) and denoted by s working with a symbiotic partner sp is given by:

$$f(s) = \frac{1}{|C|} \sum_{k \in C} c_k(\phi(s, sp)) \tag{8}$$

where C is the constraints set, $|C|$ the size of the constraints set and $\phi(s, sp)$ the induced phenotype of s when sharing information with its symbiotic partner.

It does however not give any indication of the quality of the partially defined assignments *relatively to the constraints it covers*. To see whether an association works fine or not, a restricted fitness measure is define that only considers the constraints covered by the phenotype of the solution.

Assume that $cov(s, C)$ is the set of constraints covered by s, the covering fitness measure is given by:

$$f_{cov}(s) = \frac{1}{|cov(\phi(s, sp), C)|} \sum_{k \in cov(\phi(s,sp),C)} c_k(\phi(s, sp)) \tag{9}$$

We use the first measure (8) to guide the evolutionary process (selection). The second measure is used by the evolutionary observer to decide whether a transition should be performed. The idea of introducing a mechanistic observer to decide when a transition occurs corresponds to the work in [13]. Important to remember is that although the fitness is determined using the induced phenotype the fitness value is assigned to the genotype. Hence the process of differential survival and reproduction operates at the genotype level and not at the level of the induced phenotype.

3.5 Higher-Level Individuality and Evolvability

As was assumed in the beginning of this section, solutions (genotypes) are selected according to their fitness described by (8). When a solution is selected, it will replicate into a new solution. There is a certain probability that this replication process has errors and in this way mutants can emerge.

The symbiosis between s and sp also has some consequences for the reproductive process. In certain circumstances beneficial symbiotic relations will be replicated as a whole. When the symbiotic partner is replicated as well, the symbiotic link, that is, their interaction scheme will be inherited in the process. The underlying idea is that (possibly) good working units can survive over more than one generation. The idea of performing this replication in group is based on our previous work in the context of multi-level selection [12]. For now it is assumed that decision to replicate the group is decided randomly using a probability q (here, $q = 0.5$). More elaborate methods based on the type of interaction can be used. Note that there is still a probability $(1 - q)$ of individual replication.

This replication of the both genotypes is a *first step* toward a new higher-level entity of selection. Although simple lower-level entities can sometimes replicate in group they still have the possibility of spreading their own genetic material (probability $(1 - q)$). In the *second step*, both partners give up their individual replication process in favour of a group replication process. At that point, the transition has occurred since replication becomes now the responsibility of the higher-level structure. To make this final step the function $f_{cov}(s)$, defined in the previous section, is used (see Equation (9)). When the induced phenotype happens to solve the sub-problem defined by the covering set of constraints, i.e. when f_{cov} has reached a certain threshold value (here we selected $f_{cov}(s) = 1$), a new more complex individual is created whose genotype corresponds to the induced phenotype of the previous symbiotic relation.

Let us take the example solution previously discussed. The solution described by (3) with the phenotype given in (6) has a classical fitness value of $f(s) = 0.33$. The measure of the fitness restricted to the covering constraints set was

$f_{cov}(s) = 1$. In this case, if the solution is selected, the system creates a higher level unit combining the genetic information of both partners. This means that the expression of the genotype of the new of unit is:

$$(\langle 1, 1\rangle, \langle 3, 2\rangle, \langle 2, 3\rangle) \tag{10}$$

This defines the transition step: Solutions are incrementally grown according to their success in solving the sub-problem they are defined for.

The information sharing strategy we adopted allows information to be exchanged when values are conflicting on certain allele (such as this was the case in example (5) on variable 1). This mimics a cross-over operation as full grown solution are interacting. When partial solutions interact, the exchange can only happen on conflicting parts of the genotype, yet, preserving the non conflicting parts of the genotypes. In this case, we can consider this as a preserving cross-over operation that avoids recombining good part of the solutions. In this way, the model uses the notion of compositional evolution as discussed in [5,6].

4 Identifying Transitions in Complexity

In this section, we will illustrate the transition process by analysing the outcome of a simple simulation of this model for random generated BINCSP instances of 15 variables each taking values in a domain of size 15 which are made easy or difficult by tuning parameters such as the density of the constraints network p_1 or the average tightness within the constraints $\bar{p}_2$ [11]. We propose to discuss the results for two different setups of these parameters which yield respectively a relatively easy ($\bar{p}_2 = 0.3$ and $p_1 = 0.9$) and relatively difficult ($\bar{p}_2 = 0.5$ and $p_1 = 0.5$) BINCSP instance. For each setup, we solved 25 instances and performed 10 runs for each instance. The 250 runs were then analysed by looking at the fitness relatively to the genotype size and the evolution of the size and fitness over time. Increased size of the genotype reflects successive transitions from simple units starting from length 2 up to a complex units that solves greater number of constraints.

In Figure 2, we plotted for each setup the fitness and genotype size dynamics for an isolated run, the average on all the runs of the fitness with respect to the size and the average on all the runs of the duration a genotype remains at a certain level before performing a transition.

In the first row of Figure 2 (the genotype and induced phenotype size have been rescaled to $[0, 1.0]$ for illustration purposes), a close relationship between the trend of the genotype size and the fitness trend can be observed. We can conclude from this that transitions are needed to allow the fitness to reach higher levels. For difficult problem instances, once the genotype size and the resulting phenotype size are fixed, we can observe that the fitness value still slowly improves over time. This slow improvement illustrates a phase of conflict mediation where the partial solution and its symbiotic partner try to reduce the set of variables which yield conflicts. For easy problem instances, Fitness is closely related to the genotype size, yet, we can observe that a good working symbiotic

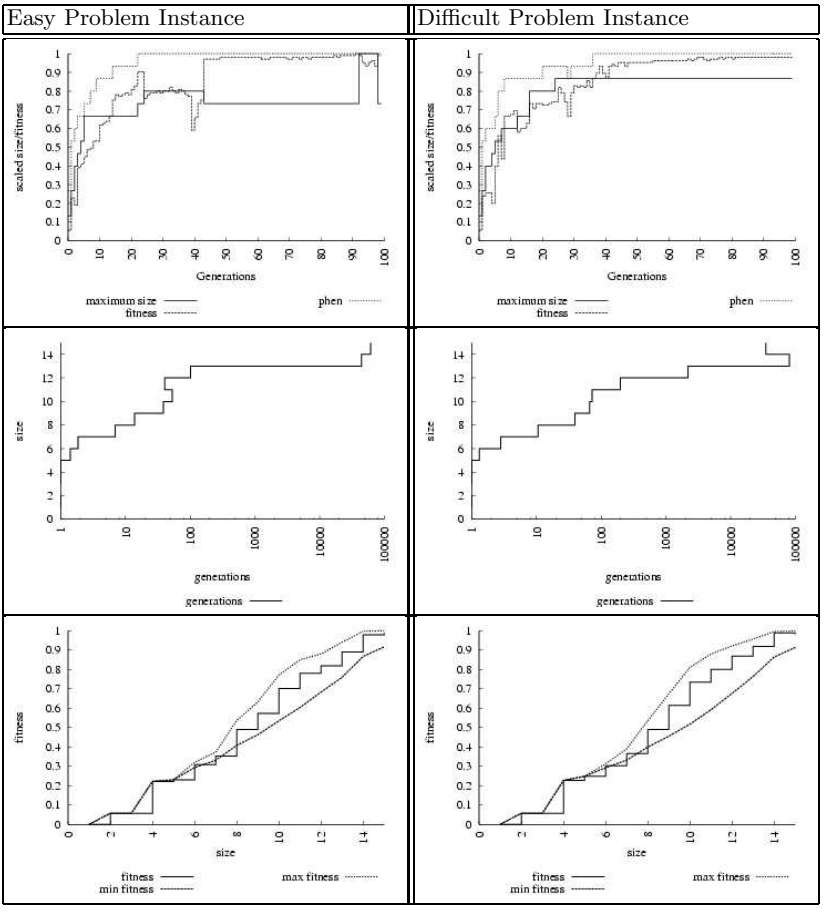


Fig. 2. The left column gives the results for an easy test case instance while the right column gives the same result for a difficult test case instance. On the first row, the evolution of the fitness relatively to the scaled genotype and phenotype size for one simulation run is plotted. On the second row, the average required time for the next transition to occur and on the last row, the fitness that corresponds to each genotype size.

relation has been discovered by the process (which can be seen on the graph by a relatively small genotype size for which the corresponding phenotype defines a complete solution). This good working collaboration is sustained for a while before the conflicts among the symbiotic partners are resolved and a transition can occur.

In the second row, which refers to the time required for different genotype sizes to evolve toward a new level of complexity, we see that the time increases as the genotypes become more complex. In other words, the conflict mediation becomes more difficult as the interacting units grow in complexity. The tran-

sitions which occurred at fast speed in the beginning require more time as the evolutionary process goes on and the conflicts to be resolved (the variables which share different values in the solution and its symbiotic partner) increase. We observe that this phenomenon is independent from the hardness of the problem instance as the increased number of generations required to master a new level of complexity is observed in both cases. Note also that this increase in time to move between complexity levels has also been observed in nature [1].

Finally, a look at the fitness relative to the genotype size (averaged over all runs) confirms the first observation that the increasing complexity at the genotype level results in an increase of the functionality of this genotype. In other words, the system requires transitions to attain the level of complexity specified by the problem instance.

5 Conclusions

In this paper, we addressed the issue of the emergence of complexity in evolutionary optimisation algorithm. Inspiration was found in the theories concerning evolutionary transitions observed in Biology. These theories propose a generalised explanation for the mechanism by which interacting lower level units can produce new higher level ones. The proposed Transition algorithm uses symbiosis as the basic ingredient for the system to work. To illustrate this model, we applied it to BINCSP and showed within this context how the mechanism of transition worked. The algorithm was also compared thoroughly to other evolutionary approaches for solving BINCSP [7]. These experiments showed the great promise for the discussed technique.

References

1. Maynard Smith, J., Szathmáry, E.: The major transitions in evolution. Oxford University Press (1995)
2. Michod, R.: Darwinian Dynamics: Evolutionary transitions in Fitness and Individuality. Princeton University Press (1999)
3. Palmer, E.M.: Graphical Evolution. John-Wiley & Sons, New York (1985)
4. Goldberg, D., Korb, B., Deb, K.: Messy genetic algorithms: motivation, analysis, and first results. Complex Systems **3** (1989) 493–530
5. Watson, R.A., Pollack, J.B.: Symbiotic combination as an alternative to sexual recombination in genetic algorithms. In: Parallel Problem Solving from Nature (PPSNVI), Springer Verlag, Lecture Notes in Computer Science 1917 (2000)
6. Watson, R.A., Pollack, J.B.: A computational model of symbiotic composition in evolutionary transitions. Biosystems Special Issue on Evolvability **69/2-3** (2002) 187–209
7. Defaweux, A., Lenaerts, T., van Hemert, J., Parent, J.: Transition models as an incremental approach for problem solving in evolutionary algorithms. In: the Genetic and Evolutionary Computation Conference, to appear (2005)
8. Tsang, E.: Foundations of Constraint Satisfaction. Academic Press Limited (1993)

9. Rossi, F., Dhar, V.: On the equivalence of constraint satisfaction problems. In Aiello, L.C., ed.: ECAI'90: Proceedings of the 9th European Conference on Artificial Intelligence, Stockholm, Pitman (1990) 550–556
10. van Hemert, J.: RandomCSP (2004) Freely available from `http://freshmeat.net/projects/randomcsp/`.
11. van Hemert, J.: Application of Evolutionary Computation to Constraints Satisfaction and Data Mining. PhD thesis, Universiteit Leiden (2002)
12. Lenaerts, T., Defaweux, A., van Remortel, P., Nowé, A.: Evolutionary game dynamics of intrademic multilevel selection. Technical Report TR/IRIDIA/2005-07, IRIDIA (2005)
13. Bersini, H.: Whatever emerges should be intrinsically useful. In: Proceedings of the ninth international conference on artificial life, MIT press (2004) 226–231

Genetic Assimilation and Canalisation in the Baldwin Effect

Rob Mills and Richard A. Watson

School of Electronics and Computer Science, University of Southampton,
Southampton, UK, SO17 1BJ
rmm101@zepler.net, raw@ecs.soton.ac.uk

Abstract. The Baldwin Effect indicates that individually learned behaviours acquired during an organism's lifetime can influence the evolutionary path taken by a population, without any direct Lamarckian transfer of traits from phenotype to genotype. Several computational studies modelling this effect have included complications that restrict its applicability. Here we present a simplified model that is used to reveal the essential mechanisms and highlight several conceptual issues that have not been clearly defined in prior literature. In particular, we suggest that canalisation and genetic assimilation, often conflated in previous studies, are separate concepts and the former is actually not required for non-heritable phenotypic variation to guide genetic variation. Additionally, learning, often considered to be essential for the Baldwin Effect, can be replaced with a more general phenotypic plasticity model. These simplifications potentially permit the Baldwin Effect to operate in much more general circumstances.

1 Introduction

Our knowledge of modern genetics suggests that an organism's lifetime adaptations cannot influence the course of evolution because learned characteristics do not change ones own genes. In the late 19th century, Baldwin argued that although a direct effect of lifetime adaptation on genes is not possible, an indirect influence on the course of evolution is [1]. Subsequently his name has been associated with the impact that learning can have upon evolution. The 'Baldwin Effect' is based on two levels of search occurring: from generation to generation we have a slow genetic variation; and within each generation the variation due to lifetime learning is faster. The combination of the two search mechanisms allows the space to be explored more completely than it would be by genetic search alone; an in-depth search around the genetically specified position is performed by the lifetime plasticity, and the genetic starting points are selected for the lifetime phenotypes they enable. This can change the selection of genotypes, providing selective gradients where none was previously available, and in particular, if the discovery of fit phenotypes during lifetime plasticity is correlated well with the genetic closeness of those genotypes to fit configurations then selection will guide evolution toward fit genotypes that may not have been discovered otherwise [2].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 353 – 362, 2005.

1.1 The Baldwin Effect

Controversy has surrounded the Baldwin Effect since Baldwin first proposed it in 1896. The hypothesis appears very similar to Lamarck's disproved beliefs that an acquired trait is directly inherited by offspring; as Turney put it, "*Baldwinian and Lamarckian evolution are virtually indistinguishable in their effect"* [3]. However, unlike Lamarckian evolution, the Baldwin Effect is compatible with genetics since it does not require the direct inheritance of acquired characteristics. There is perhaps a little irony here in the debate over Baldwin's 'organic selection' hypothesis since it was meant to replace Lamarck's failed theory [4-5].

In 1987 Hinton and Nowlan published the first computational model [6] to demonstrate the Baldwin Effect which provides excellent insight into how the effect works. They use a simple model where a population is given a 'needle on a plateau' problem with a single phenotype having increased fitness from an otherwise equally fit plateau. A string with 20 'genes' of 0's, 1's, and ?'s (in initial frequencies 0.25, 0.25, 0.5, respectively) is used to represent each individual's genotype; 0's and 1's represent genetically specified traits, and ?'s represent phenotypically plastic traits. A population of 1000 individuals are randomly initialised. The population is evolved using one-point crossover between two parents selected proportional to their fitness. Within each lifetime, each organism completes 1000 lifetime learning trials, where loci with ?'s in the genotype are randomly assigned a new allele of 0 or 1. The all 1's phenotype is the fitter combination; the fitness of an organism is proportional to the number of lifetime trials left after the all 1's phenotype is found. Thus, a genotype with no 0's may reach the peak in the fitness landscape, and the more 1's it has the more likely its phenotypes are to hit that peak. So when an individual finds this peak in a phenotypic trial, it obtains a greater fitness and begins to dominate the population quickly removing individuals containing 0's. Once there exists a number of individuals who can all reach the peak in their lifetime, the selection pressure shifts to differentiate between these individuals. An individual who is genetically closer to the peak will more reliably hit the peak during its lifetime. In this way, exhibition of the good trait in the genotype is selected for. In typical runs of the simulation after very few generations, the all 1's phenotype is found by lifetime learning, and the average number of 1's in the genotype increases rapidly. In subsequent generations the average number of 1's in the genotype increases slowly towards the fittest genotype.

When a comparable population without lifetime learning is simulated, the all 1's phenotype takes an unreasonable time to be found and there is thus no pressure to increase the number of 1's in the genotype. From this result we can see that the presence of lifetime learning can influence the selective pressure for genetic traits; a learned trait can be genetically assimilated without any direct genetic transfer from phenotype to genotype. A second effect, canalisation, is also exhibited by such models of the Baldwin Effect. Canalisation, or reduction in lifetime plasticity, is facilitated by means of reduction in numbers of ?'s – the allele representing that plasticity. The reduction of ?'s only begins to occur after all-1's phenotypes have been discovered and the 0's have been removed from the population. Selection favours those who find the all-1's phenotype more quickly over those that find it more slowly, and in the Hinton and Nowlan model, the only way to achieve this is a reduction in ?'s and hence canalisation. Indeed, it may be viewed that an individual's genetically specified traits are preferred in this model because it requires less lifetime learning –

implying that canalisation is a necessary part of the Baldwin Effect. This is not correct. In general, a genotype may be selected for in the Baldwin Effect because it produces better phenotypes and, as we will show, this need not necessarily imply that it has less variation in phenotypes as it does in the Hinton and Nowlan model.

Several research papers have been inspired by the work of Hinton and Nowlan (H&N), on a variety of topics. H&N's model is analysed in [7-9], which all focus on the canalisation effect in the experiment; that is pressures for the reduction in plasticity following a learned behaviour becoming genetic. A different kind of evolutionary simulation is used in [10] to demonstrate the Baldwin Effect; they use a simulated physical world in which the population size is variable, and has food costs associated with reproduction, movement and metabolism. Fitter phenotypes benefit an organism in reduced costs for one of the actions. Watson and Pollack adapt H&N's model to demonstrate how symbiosis can produce organisms which would not have evolved without the support of a symbiont [11]. This work is extended to show that in a sparse ecosystem, when one of the symbionts can perform the task independently, the symbiont can offer no advantages and thus becomes a parasite [12]. Another study presents a cultural model with learned and inherited behaviours, using a physical world similar to [10], but with shared behaviours between phenotypes [13]. Results are similar to [11-12]; when behaviours are shared in abundance, assimilation advantage (i.e. selection for independence) reduces. Turney identifies confusion surrounding Baldwinian and Lamarckian evolution, and highlights that although the benefits of learning are demonstrated, often the costs of learning pass without acknowledgement. A series of experiments on drosophila demonstrate 'genetic assimilation' towards wings without cross-veins following a temperature shock [14].

However, the Baldwin Effect does not appeal to all researchers: [15] and [16] both write that the effect is not of much interest or importance in evolution and that Baldwin himself was not primarily interested in 'organic selection'; instead, social heredity or niche construction should be the subject of further study.

Some relevant summary points are made by Turney:

1. Lifetime learning smoothes the fitness landscape since the phenotypic exploration is local to its inherited genotype.
2. There are benefits to phenotypic rigidity: it is advantageous to evolve some rigid mechanism to replace learned mechanisms over time, since learning requires experimentation (for example, learning how to hunt could be dangerous).

Though both these points may often be true in natural populations, and Hinton and Nowlan's model includes both, they are quite separate issues. Reviewing the literature on the Baldwin Effect and phenotypic plasticity in general, a question is raised regarding the difference between genetic assimilation [14] and canalisation [17]. Only a subtle distinction exists and we find no previous model that attempts to show the Baldwin Effect without canalisation, i.e. reduction in plasticity, or discussion that identifies this distinction. Thus, we propose to use the words as follows: canalisation is a reduction in phenotypic plasticity; genetic assimilation occurs when a behaviour that was once acquired in the phenotype becomes specified in the genotype. The conceptual distinction is easily recognised by considering how the mean and variance of the distribution of phenotypes changes over evolutionary time: canalisation means

that the variance in phenotypes reduces, genetic assimilation means that the mean phenotype is moved but does not necessarily suggest that the width of that distribution might reduce. Thus we suggest that genetic assimilation, the key aspect of the Baldwin Effect, does not logically require canalisation, i.e. the phenotype to reduce its level of plasticity.

Note that H&N's model shows both a change in the mean and the variance of phenotypes, i.e. genetic assimilation and canalisation. In the following section we provide a simplified adaptation of H&N's model to show just genetic assimilation without canalisation in order to separate these concepts and illustrate that the Baldwin Effect (being essentially genetic assimilation) does not require canalisation. Since an organism with plasticity (in a particular activity) will still benefit from having that trait genetically well-adapted; it has more chance of an appropriate phenotype occurring than if the genotype is poorly adapted. We will see this demonstrated in the first experiment in section 3.

2 Constant Plasticity Model

The work of Hinton and Nowlan as described above provides a simple model which demonstrates the Baldwin Effect. However, some features of the model are not required to show the Baldwin Effect, and here we present a simpler model. This simplification aims to reduce the assumptions required and realise a model which could be applied to a more general set of cases, and also to assist understanding, specifically to separate the concepts of genetic assimilation and canalisation. The main issues addressed by these simplifications concern a learning model that recognises successful phenotypes, the conflation of canalisation and genetic assimilation, and the mismatch of genetic and phenotypic variation spaces.

H&N use a lifetime plasticity model that involves recognising when a good phenotype is discovered (which may be called learning). If the mechanism of the Baldwin Effect derives simply from smoothing the fitness landscape, as Turney suggests, then a simpler more direct model of lifetime plasticity should suffice such as the average fitness of random phenotypic variants. To remove the assumption of learning phenotypes, we evaluate fitness as the mean fitness of the lifetime phenotypes, rather than the number of trials remaining after the phenotypic solution is first found. This means that the organisms do not have to recognise their own success (as is implicit in H&N's model).

As established in section 1, H&N's model contains both genetic assimilation and canalisation. We propose that this canalisation is an unnecessary element to display the Baldwin Effect, and as such a significant change to the model is required. In our constant plasticity model there is no designation of particular traits that are phenotypically plastic, i.e. we do not use '?' alleles. Instead lifetime plasticity varies any trait with equal probability, using non-heritable mutation-like variations applied to the genotypic trait specifications in each lifetime trial. This models a constant plasticity level which separates the effects of lifetime plasticity from canalisation.

It has been suggested that in order to enable genetic variation to follow lifetime variation it is desirable to have genetic variations and lifetime plasticity using the same or correlated variation operators [2]. Thus, we can simplify the model further by facilitating genotypic variation using the same method as the phenotypic variation

in lifetime trials, i.e. by spontaneous point mutation, instead of using sexual recombination.

These three key changes to H&N's model produce one which is considerably simplified: a population of binary strings is reproduced with fitness proportionate selection and mutational variation. The fitness of each individual is the average fitness of the phenotypes it produces during its lifetime and the fitness of each phenotype is F_{max} if it is all 1's and 1 otherwise. Each phenotype is a random mutation-like variation of the genetically specified traits (rather than a random filling-in of a variable number of ?') This model is described in more detail below:

```
1) Initialise population of P individuals,
2) for each generation
   a) for all individuals, i:
      i)  for each lifetime trial, 1 to L:
           (1) generate phenotype by adding random variation
           (2) evaluate fitness
      ii) calculate mean fitness across all phenotypes for individual i
   b) select P parents with probability proportional to fitness
   c) generate offspring by mutating the parents genotype
```

The number of mutations in each lifetime trial (and each genetic reproduction) is governed by an exponentially decaying distribution in which it is most likely for no mutations to occur, but some small probability of a large mutation count exists. The probability of a given phenotypic trial (or evolutionary step) having k mutations is given by $P(m=k) = exp[-\alpha k/N] \cdot (1-exp[-\alpha/N])$, where N is the number of traits, and α is a rate parameter. At each loci marked for mutation, a new random allele is produced with equal probability of 0 and 1. This is used in order to allow the possibility of a large number of mutations, whilst maintaining the condition that the most likely phenotype will be identical to its genotype (this is not true of the usual Poisson distribution of mutation counts which results from common mutational models).

3 Simulation Experiments

We simulate populations in a variety of configurations firstly to demonstrate that a population with a constant lifetime plasticity level can manifest the Baldwin Effect (and that populations without lifetime plasticity do not show the effect), and secondly to aid explanations which distinguish genetic assimilation from canalisation.

The evolutionary algorithm used follows Hinton and Nowlan's model, with the modifications as detailed in section 2. The parameter values can be summarised as follows:

Population size, P:	2^{10}
Lifetime trials, L:	2^{10}
Number of traits, N:	20
Fitness of best phenotype, F_{max}:	1000
Phenotypic mutation parameter, α_p:	5
Genotypic mutation parameter, α_g:	9.1

For the non-plastic experiment, all parameters are as above except that there are no phenotypic trials, i.e. the evaluated phenotype is identical to the genotype. A parallel pair of experiments using Hinton and Nowlan's model are also run; in the non-learning H&N population the ?'s are not used and all variation is performed by sexual crossover in the genotype. Figure 1 shows one typical run of the constant plasticity model, and one run of H&N's model is depicted in figure 2. Due to the stochastic nature of the effect, a single typical run is more informative than mean values over many runs.

Figures 1(a) and 2(a) show data which reveal the mean composition of the population. By calculating the mean distance from the consensus phenotype across all phenotypic trials in each individual and taking the mean of these, we provide a measure of the mean number of phenotypic variations per individual, which is shown by the dotted line. This is a more suitable measure of phenotypic variations than variance of the number of 1's in the phenotypes of an individual, since the value is not dependent on the position in the phenotype space, whereas the variance of the number of 1's is dependent on the mean number of 1's of that particular phenotypic distribution. The dot-dashed line shows the mean number of 1's in the non-plastic population; 100 generations are shown but the behaviour continues similarly for several hundred generations; all non-plastic runs take longer than all plastic runs, with a mean of greater than 1600 generations to find the fitter solution (although some fortunate runs do succeed without lifetime plasticity in much less than this). Figures 1(b) and 2(b) show the progression of distributions of phenotypes across the population through the experiments with plasticity or learning.

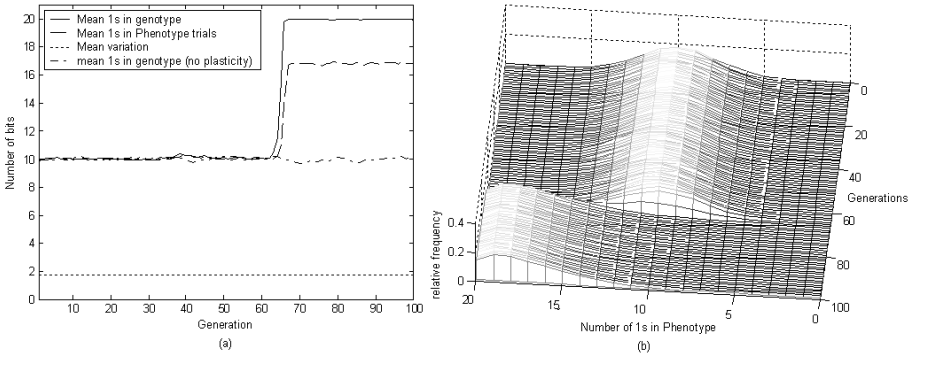


Fig. 1. Typical behaviour of the constant plasticity model

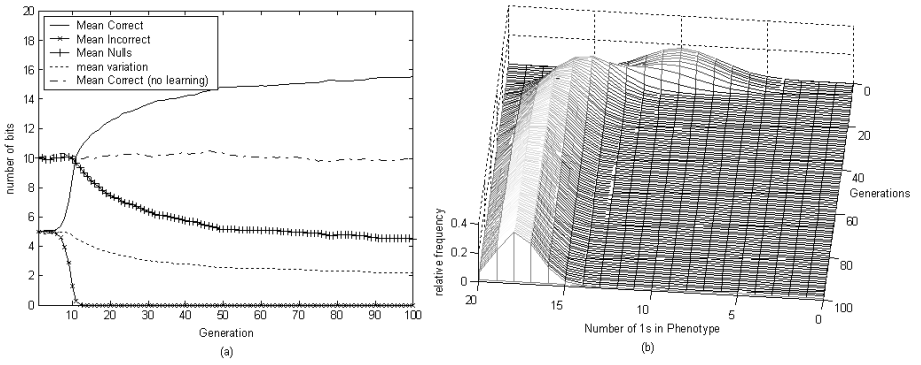


Fig. 2. Typical behaviour of Hinton and Nowlan's model

4 Discussion

Here we have presented a model which shows that although already simple, H&N's model has elements unnecessary to manifest the Baldwin Effect. Specifically, the results for our model show that lifetime plasticity can accelerate evolutionary search without: 1) the need for subsequent reduction in lifetime plasticity, 2) a learning model that recognises success, or 3) sexual recombination.

If we first consider the results in figure 1(a), it takes approximately 60 generations before the selective pressure favouring genotypes with more 1's overcomes stochastic effects (genetic drift) and is able to affect genotype frequencies. When this happens the population finds the fittest genotype very shortly afterwards. The transition is really too rapid to see that genotypes with more than 50% 1's *but not all 1's* increase in frequency before the genotypes with all 1's appear, but we can see that this pressure is present because in the case without plasticity the mean number of 1's in the genotype does not increase at all. Part of the reason for the rapidity of this transition, as compared to the H&N model, is that since genetic crossover is not used, the population does not have to wait for two fitter individuals to 'find' each other before the fitter genotype can proliferate. The fittest genotype dominates the population which is indicated by the mean number of 1's in the genotype being very close to 20. The mean number of ones in the phenotype is lower since mutations still occur (in both the phenotype and the genotype) after genetic assimilation has taken place.

Results for non-plastic populations in figures 1(a) and 2(a) (the dot-dashed lines), show that both the non-plastic cases have approximately constant values for mean number of 1's in the genotype. In 30 additional runs this continued for a mean of over 1600 generations for the CP model, and a mean of over 3000 generations for the H&N model.

The dotted line in figures 1(a) and 2(a) provides a measure of the variation occurring in the phenotype from its genotype. This mean value of variation is unsurprisingly constant in the constant plasticity model, both before and after the fitter genotype has dominated the population. However, this is not the case in H&N's model; here the variation decreases through the course of the experiment. Specifically, the decrease in this variation is proportional to the mean number of ?'s present in the

population's genotypes; the ?'s represent plasticity (or learning ability) so the relationship should be expected. More information is purveyed in figures 1(b) and 2(b) which depict the phenotypic distribution for these two experiments. Figure 1(b) shows a change in the mean of this distribution from approximately 10 to approximately 17 around generations 60-65; this is the point of genetic assimilation. (The mean values are also plotted on the dashed line in figure 1(a)). However, the width (variance) of this distribution is constant both before and after genetic assimilation has occurred in our model. The distribution progression shown in figure 2(b) has a different behaviour. The mean number of this distribution is again initially at 10 bits; as the 0's are purged and replaced with 1's the mean moves upwards; the distribution width is constant as the number of ?'s is also approximately constant. However, as the selection pressure begins to replace ?'s with 1's, both the mean and variance of the phenotypic distribution are changed. The mean is increased further as more 1's exist; since they replace ?'s the learning ability is reduced which directly reduces the variance. Thus, figure 2(b) shows first genetic assimilation – the shift in mean, followed by canalisation – the narrowing of the ridge, or reduction in learning ability. Since the mean of both distributions moves towards 20 1's, variation can only reduce the number of 1's present, so the shape of the distribution becomes skewed; this is an unfortunate artefact of compressing 20-dimensional data to a single axis. The variance is thus also plotted on the dotted line in figures 1(a) and 2(a), which shows these trends more clearly. We have demonstrated the cases of genetic assimilation (a change in mean) and canalisation (a reduction in variance) together as per H&N's model, as well as genetic assimilation independently in our simplified model. It is easy to imagine a third situation in which canalisation occurs without genetic assimilation; the variation of phenotypes would reduce about the mean, but the mean would be unaffected.

As already mentioned, a single typical run for each model is shown; however one point to note is that the specific number of generations required for the fitter genotype to be found in the constant plasticity model varies from run to run more than the number of generations required before the ?'s begin to be purged in H&N's model. This may be because incorrect alleles (0's) in the H&N model occur with half the probability that they do in the CP model. This relates to another issue with the CP model. The mechanism governing phenotypic variation in the CP model allows variations to occur on any trait whereas phenotypic variation in the H&N model only allows variation to occur on specified plastic traits. Since the plastic traits are exactly the traits that are not yet correct this gives the H&N model a distinct advantage with respect to the probability that mutations will occur in the 'correct' loci for an individual to gain fitness. Arguably, this restricts the ability of genetic assimilation without canalisation, but it also causes us to question the validity of the assumptions used in the H&N model.

5 Conclusion

The Baldwin Effect is investigated and a key ambiguity in current literature between genetic assimilation and canalisation is identified. A new model is presented which is simpler than Hinton and Nowlan's landmark model, yet still manifests the Baldwin

Effect. Specifically, this new model does not use canalisation, individuals do not have to recognise their own success (i.e. cognitive learning is not required, only some form of phenotypic plasticity), and it unifies the genetic and phenotypic variation mechanisms. This shows that canalisation and learning, generally considered to be intrinsic features of the Baldwin Effect, are in fact not necessary to show that non-heritable phenotypic variation can guide genetic variation. Providing a simpler model assists us in revealing the essential mechanisms involved. These simplifications also widen the scope in which the Baldwin Effect can be applied by reducing the assumptions necessary for the effect: it may still guide the course of evolution in situations where a mechanism for canalisation is unavailable, in organisms or systems which are adaptable but not intelligently so, and in asexually reproducing populations.

Simulated experiments demonstrate the difference between the genetic assimilation and canalisation components which are often unnecessarily conflated, by consideration of the mean and variance of the distribution of phenotypes produced in a population. The conditions can be summarised as follows: canalisation occurs when there is a reduction in the variance of the phenotypes (but not necessarily a movement in the mean phenotype), whereas genetic assimilation produces a movement in the mean phenotype (but does not necessitate a reduction in the variance of phenotypes).

References

1. Baldwin, J.M.: *A New Factor In Evolution*, American Naturalist 30 (1896) 441-457, 536-554
2. Mayley, G.: *Explorations into the interactions between learning and evolution using algorithms,* DPhil Thesis, School of Cognitive and Computing Sciences, University of Sussex (2000)
3. Turney, P.: *Myths and Legends of the Baldwin Effect*, Proceedings of the 13th International Conference on Machine Learning (1996) 135–142
4. Richards, R.J.: *Darwin and the Emergence of Evolutionary Theories of Mind and Behavior*, Chicago: University of Chicago Press (1987)
5. Dennet, D.C.: *Darwin's Dangerous Idea: Evolution and the Meanings of Life,* London, Penguin (1995)
6. Hinton, G.E., Nowlan S.J., *How learning can guide evolution,* Adaptive individuals in evolving populations: models and algorithms, Addison-Wesley Longman (1987) 447-454
7. Belew, R.K.: *When both individuals and populations search*, In Schaffer, J.D. (ed.) Proceedings of the Third International Conference on Genetic Algorithms, San Mateo, California: Morgan Kaufmann (1989)
8. Harvey I.: *The puzzle of the persistent question marks: A case study of genetic drift*. In S. Forrest (editor) Proceedings of the Fifth International Conference on Genetic Algorithms, ICGA-93, California: Morgan Kaufmann (1993)
9. Wiles, J., Schultz, R., Hallinan, J., Bolland, S., Tonkes, B.: *Probing the persistent question marks,* In L. Spector, E. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon & E. Burke (Eds), Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001). San Francisco, CA: Morgan Kaufmann Publishers (2001) 710–717

10. French R.M., Messinger A.: *Genes, phenes and the Baldwin Effect: Learning and evolution in a simulated population,* Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, Brooks, R. A. and Maes, P. (Eds.) MIT (1994) 277-282
11. Watson, R.A., Pollack, J.B.: *How Symbiosis can guide Evolution,* Fifth European Conference on Artificial Life. Dario Floreano, Jean-Daniel Nicoud, Francesco Mondada, eds. Springer (1999)
12. Watson, R.A., Reil, T., Pollack, J.B.: *Mutualism, Parasitism, and Evolutionary Adaptation*, Proceedings of Artificial Life VII, Bedau, M, McCaskill, J, Packard, N, Rasmussen, S (eds.) (2000)
13. Federici, D.: *Culture and the Baldwin Effect*, Lecture Notes in Computer Science 2801 (2003) 309-318
14. Waddington, C.H.: *Genetic Assimilation of an Acquired Character*, Evolution 4 (1953) 118-126
15. Simpson G.G.: *The Baldwin Effect,* Evolution 7 (1953) 110-117
16. Griffiths P.E.: *Beyond the Baldwin Effect: James Mark Baldwin's 'social heredity', epigenetic inheritance and niche-construction'*, in Learning, Meaning and Emergence: Possible Baldwinian Mechanisms in the Co-Evolution of Mind and Language, Weber, B. and Depew, D. (eds), (2003)
17. Gibson, G., Wagner, G.: *Canalization in evolutionary genetics: a stabilizing theory?*, BioEsssays 22:, John Wiley (2000) 372-380

How Do Evolved Digital Logic Circuits Generalise Successfully?

Simon McGregor

Centre for Computational
Neuroscience and Robotics, UK
sm66@sussex.ac.uk

Abstract. Contrary to indications made by prior researchers, digital logic circuits designed by artificial evolution to perform binary arithmetic tasks can generalise on inputs which were not seen during evolution. This phenomenon is demonstrated experimentally and speculatively explained in terms of the regular structure of binary arithmetic tasks and the nonoptimality of random circuits. This explanation rests on an assumption that evolution is relatively unbiased in its exploration of circuit space. Further experimental data is provided to support the proposed explanation.

1 Introduction

This paper deals specifically with the generalisation ability of artificially evolved digital logic circuits, a subject which has so far received little attention. I report an empirical phenomenon which gives rise to interesting theoretical conjectures, some of rather wide relevance. The reader is assumed to be familiar with Boolean algebra; for readers unfamiliar with the fields of artificial evolution or digital logic circuits, brief descriptions are given.

Sections 2-4 of the paper provide general background, details of the target problems and training method. Experimental results on the generalisation performance of evolved circuits on 7 different binary arithmetic problems are presented in section 5. These results, with one notable exception, show unambiguously that the circuits generalise at a better-than-chance level. Section 6 proposes an explanation based on complexity-related ideas; some modest empirical support for this explanation is described in section 7.

2 Background

2.1 Artificial Evolution

The use of techniques inspired by biological evolution for design and optimisation tasks has become commonplace. These techniques, known collectively as *artificial evolution* [10], are now so well-known that I will not describe them in detail. The basic principle is that one can find a solution to a problem by starting with

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 363–372, 2005.

a randomly generated initial population of candidate solutions and applying various *genetic operators* to them to produce new solutions. Candidate solutions are evaluated according to a *fitness function* and, over time, worse solutions are replaced by better ones. The process is repeated until either a satisfactory solution is found or the experimenter terminates the search.

2.2 Digital Logic Evolution

A number of researchers e.g. [3,8,11] have applied artificial evolution to the design of Boolean logic circuits. These are circuits made out of binary logic gates such as AND, OR or NOT gates. Such research is often regarded as belonging to the field of *evolvable hardware*, but Boolean circuits can also be regarded as pure mathematical entities (combinational circuits). The circuits described in this paper are composed of 2-input gates connected acyclically.

2.3 Generalisation in Evolved Logic Circuits

Existing work on evolved digital logic circuits has tended to focus on circuit optimisation [7], discovery of new design principles [9] or has used the domain as a convenient test bed for new evolutionary methodologies e.g. [11,3,5]. Consequently, exploration of the generalisation properties of evolved circuits has been within the context of speeding up evolutionary search. Miller & Thomson [9] looked at generalisation performance when a small proportion ($< 50\%$) of rows were presented, and found that evolution did not generate functionally perfect solutions. In concluding that evolved circuits did not generalise, however, they did not consider whether the circuits performed better than chance guessing.

Imamura et al. [2] provide a theoretical analysis also focussing on 100% correct circuits. For an evolved circuit with O outputs, which is correct on T training samples out of N possible inputs, they derive a probability of $2^{-O(N-T)}$ that the circuit will generalise correctly on the entire possible input set. However, their argument rests on the assumption that the output for untrained input data is a uniformly distributed random variable, which I will show experimentally does not hold for several binary arithmetic functions.

Clearly, no algorithm can generalise on all possible problems. For every truth table which is similar to a circuit's output, there is by definition another which is dissimilar to an identical magnitude. Therefore, considered over all possible truth tables consistent with the training vectors, any algorithm will produce circuits whose guesses are on average precisely 50% correct (i.e. at chance level).

So the question is, given a set of training vectors (e.g. 75% of the rows of a binary multiplication table), what sort of truth tables do evolved circuits tend to generalise to? Are they evenly distributed across the available function space, or do they tend to cluster in particular areas? I present empirical results showing that for the problems considered, they disproportionately implement human-recognisable binary arithmetic tables.

Table 1. Truth tables used as target functions for digital logic circuit evolution, along with the CGP geometry and maximum number of generations during evolution

Table	Geometry	Max Gens	#In	#Out	Description
1-Bit Adder	1 x 30	20K	3	2	1-Bit plus 1-Bit Binary Adder with Carry
2-Bit Multiplier	1 x 40	20K	4	4	2-Bit by 2-Bit Binary Multiplier
2-Bit Adder	1 x 40	20K	5	3	2-Bit plus 2-Bit Binary Adder with Carry
2.5-Bit Multiplier	1 x 50	200K	5	5	3-Bit by 2-Bit Binary Multiplier
2.5-Bit Divider	1 x 50	500K	5	5	3-Bit by 2-Bit Binary Protected Divider with Remainder (x/0 was defined as 0 rem 0)
Even 6-Bit Parity	1 x 30	30K	6	1	Even 6-Bit Parity
3-Bit Multiplier	1 x 50	10M	6	6	3-Bit by 3-Bit Binary Multiplier

N.B. Most studies on the even n-parity problem explicitly disallow the use of XOR or XNOR primitive gates to force evolution to build these gates out of other gates. However, in the experiments reported here evolution was free to use any 2-input gate (including XOR and XNOR) in evolving even 6-parity.

3 Target Problems

Circuits were evolved to perform basic binary arithmetic functions: the inputs to a circuit are interpreted as two binary numbers, and the outputs are interpreted as one or more binary numbers which are some arithmetic function of the input. For example, a *2-bit multiplier* is a circuit with 4 binary inputs (representing two 2-bit binary numbers) and 4 binary outputs (representing one 4-bit binary number). Binary adders incorporate a 1-bit *carry* input and provide a 1-bit *carry* output as well. Circuits were also evolved to be *even n-parity* calculators. This type of circuit has n binary inputs and 1 output; the output is *true* if the total number of *true* inputs is even, and *false* otherwise.

A feed-forward binary logic circuit can be described by a *truth table*, which exhaustively lists the desired circuit outputs for each possible combination of inputs. Hence, a truth table has 2^n rows where n is the number of circuit inputs. The truth tables used were as shown in table 1.

4 Training Method

4.1 Training Vectors

For each truth table, a variety of training sets were generated by randomly (uniformly) *ablating* a certain proportion of output bits, i.e. setting them to "don't care". Two protocols were used: *row ablation*, in which entire rows of the truth table output were ablated, and *bit ablation*, in which individual output bits were ablated. The algorithm used produced an exact number of ablations rather than merely ablating rows or bits with a particular probability.

4.2 Circuit Evolution

Circuits were evolved under a 1 + 1 scheme with adaptive mutation rate (following [1]) and no crossover. This type of algorithm, effectively a hill-climber with neutral exploration, generates a single mutant of the current-best individual each generation and replaces the current-best individual with the mutant if the mutant is of equal or greater fitness (measured by training error). A *Cartesian Genetic Programming* (CGP) encoding [8] was used - this is a variable-length encoding for combinational circuits. A novel genetic operator, the *plagiarism* operator [6], was used to reduce the number of fitness evaluations required to solution. Specific details of the algorithm and encoding can be found in [6], although the results reported here should be replicable under different evolutionary schemes. During evolution, the circuits were evolved to minimise their error on a particular training set; the circuits' outputs for the "don't care" entries in the ablated truth table were ignored. All evolutionary runs were able to produce a circuit which achieved 100% performance on the training set. It should be noted that there was no explicit pressure for evolution to produce small circuit sizes.

4.3 Generalisation Testing

When a circuit was found which achieved zero training error, it was then tested on the "don't care" outputs for the training set. The target values for these bits were taken from the actual values of these bits in the full (non-ablated) truth table. Generalisation error was defined as the total number of incorrect output bits divided by the number of ablated bits in the test set, i.e. the mean output error per bit. In principle, evolution was free to produce circuits which gave completely arbitrary results for these "don't care" bits, in which case the expected generalisation error would be 0.5.

5 Generalisation Performance

For each truth table, 100 training sets were generated for each of 10 different ablation proportions[1] and each of 2 different ablation protocols[2] (*row* and *bit* ablation). Since the training sets were randomly generated, some of them could have been repeats. On every such training set, a zero training error circuit was evolved as described above and its generalisation error evaluated.

The results are displayed in figure 1. Each graph shows the mean generalisation error over 100 trials for the various problems, as it varies with differing proportions of ablation. The upper and lower limits for statistical significance at the p=0.05 level are indicated (derived from the normal approximation to a

[1] Except for the 1-bit adder, which has only 8 rows in its truth table and which was tested on 8 different ablation proportions.

[2] Except for the 3-bit multiplier, which was tested only on row ablation due to computational expense, and the even 6-parity calculator, which only has 1 output bit per row, making the two conditions identical.

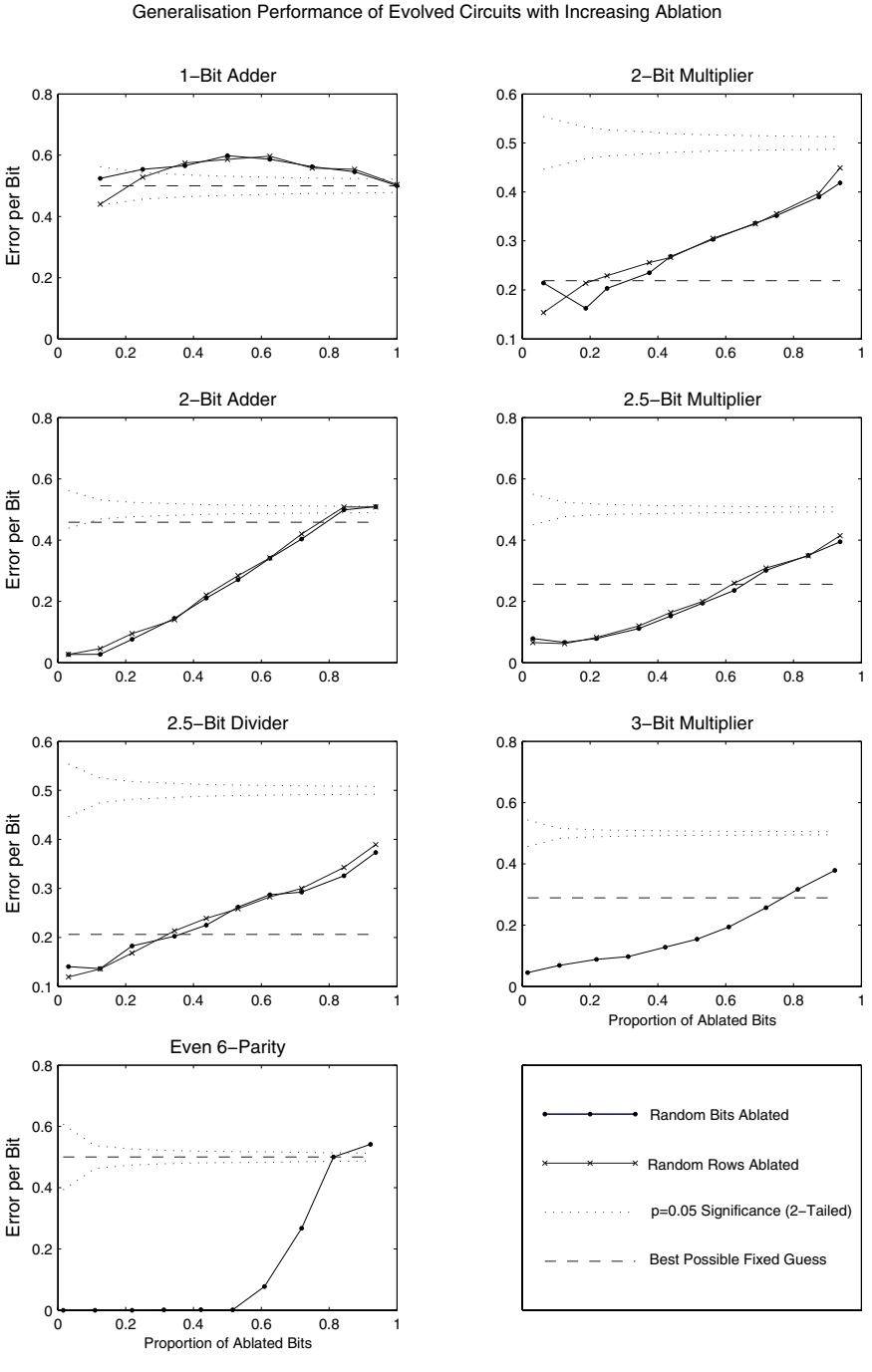


Fig. 1. Generalisation performance with increasing ablation for 7 binary arithmetic problems

binomial distribution). A line ('Best Fixed Guess') showing the least mean error which would result from constant guessing either one or zero is also shown.

The results show that the 1-bit adder is different from the other problems. This fact is discussed later in this document and proves to be highly instructive as to the mechanism which underlies generalisation.

Secondly, it is clear that evolved circuits generalise at a better than chance level on all the other problems. This holds even up to ablation levels of 80% or greater, and is independent of the overall ratio of ones to zeros in the problem's truth table. For both addition and multiplication, generalisation performance of functionally correct circuits improves with the size of the problem. For the "even 6-bit parity" problem, 99% of evolved circuits were functionally perfect parity calculators when evolved on 50% of the truth table. According to Imamura et al.'s argument [2] (see section 2.3), only $\frac{1}{2^{32}}$ of evolved circuits should have been functionally perfect.

6 Proposed Explanation

The proposed explanation rests on the notion of the *gate complexity* of a function, which is the minimum number of 2-input logic gates (from the 16 possible 2-input Boolean functions) required to implement that function in a circuit. This measure is analogous to the notion of Kolmogorov complexity [4] for strings. I define the *effective size* of a circuit to be the gate complexity of the function it implements, i.e. the gate count of the smallest functionally equivalent circuit.

Three simplifying assumptions are sufficient to provide a plausible account for the observed generalisation capacity of evolved digital logic circuits. These assumptions are sketched out below and then discussed individually in detail.

1. *Inverse relation of redundancy to frequency.* The effective gate count of a randomly chosen circuit with given input-output behaviour is more likely to be small than large.
2. *Regularity of arithmetic.* Boolean arithmetic functions are more compressible than their near variants.
3. *Lack of evolutionary bias.* The size-n evolved circuits are drawn approximately uniformly from the set of all possible size-n solutions.

6.1 Inverse Relation of Redundancy to Frequency

Most possible circuits are not optimally compact; that is to say, their effective size is smaller than their actual size. Consider the set S_n of all possible circuits of size n. It seems plausible that few of these circuits will be optimal, and that there are more ways in which a circuit can be largely redundant than slightly redundant. In other words, the number a of circuits in S_n with effective gate count x should tend to decrease monotonically with increasing x until a reaches zero. It also seems plausible that the same reasoning will apply to the set $S_{n,f}$ of possible circuits of size n which correctly implement the partial function f (i.e.

the set of size-n circuits with 0% error on the training set represented by f). That is, a circuit chosen (uniformly) randomly from this set will be more likely to have an effective size of x than an effective size of y iff $(x \leq n \wedge x < y)$.

6.2 Regularity of Arithmetic

If the target functions were random, the generalisation results observed here would not be possible, and Imamura et al.'s argument [2] of the impossibility of generalisation for evolved circuits would hold. But intuitively, we would expect the truth tables of binary arithmetic functions to be highly regular, i.e. compressible. For instance, the even 6-parity calculator has very low gate complexity; a variant of this function in which one row's output was flipped from 1 to 0 would require several more gates to implement. I present some empirical evidence in section 7 on the actual compressibility of binary arithmetic functions compared to their near variants.

6.3 Lack of Evolutionary Bias

The two previous assumptions are sufficient to explain how a random search of circuits would be expected to produce better-than-chance generalisation performance. In fact, even though it takes pseudo-random decisions, artificial evolution is not perfectly unbiased, as the structure of the *fitness space* strongly affects the path that evolution takes through it to a final solution. But since there is currently no positive reason to suppose that a special evolutionary mechanism underlies the generalisation capacity of artificial evolution, my default assumption is that evolution is more or less doing the same job as random search. Further experiments comparing the generalisation performance of circuits found by random search and evolution could confirm or disconfirm this.

6.4 Putting the Argument Together

If these three assumptions are true, then the explanation works as follows: - the near variants of a binary arithmetic function are by assumption 2 more complex (in gate complexity terms) than the function itself. Other things being equal, binary arithmetic functions are by assumption 1 more numerous amongst circuits of a given size[3]. Consequently, by assumption 3, evolution will find them more frequently than it finds their variants. The next section contains results which I hope will make this explanation seem plausible.

7 Further Experiments

The actual gate complexity of a Boolean function is expensive to compute; infeasibly so for most circuits. In this section I use the minimum evolved size as a proxy for the actual gate complexity; for the 1-bit adder, the problem is small enough to make it likely that these figures reflect the true gate complexity.

[3] Providing, of course, that size exceeds the gate complexity of the function.

7.1 Difficult Rows and Favoured Errors

Experiments were run on the 1-bit adder, 2-bit adder, 2-bit multiplier and 2.5-bit multiplier to determine what solutions evolution found when single rows of the truth table were ablated (set to "don't care"). Due to symmetry, some rows are equivalent to other rows (e.g. the rows $1 + 0 + 0$, $0 + 1 + 0$ and $0 + 0 + 1$ in the 1-bit adder); only one row was considered for each equivalence class. For every such row of each of these 4 truth tables, 100 evolutionary runs were performed with that row ablated and all other rows intact. The resulting circuits were tested on the missing row and their output recorded. This produced a distribution of completions for the missing row. For most rows, the most frequent completion was the arithmetically correct completion, although this information was not available to the algorithm during evolution.

A *difficult row* was defined as a row where the most frequent completion was arithmetically incorrect (e.g. $0 + 0 + 1 = 3$ for the 1-bit adder); a *favoured error* was the most frequent completion for a difficult row. For each favoured error, a new truth table was formed by replacing the ablated row with the incorrect completion, and 300 evolutionary runs were performed on the new truth table to estimate its minimum gate complexity.

Table 2. "Difficult Rows & Favoured Errors": number of rows with arithmetically incorrect most-frequent completions when ablated; mean smallest evolved circuit size for these completions; smallest evolved circuit size for arithmetically correct function

Function	Difficult Rows	Total Rows	Mean Min. Size (Favoured Errors)	Min. Size (Correct Circuit)
One-Bit Adder	8	8	4	5
Two-Bit Adder	0	8	N / A	10
Two-Bit Multiplier	3	16	6.67	7
3-Bit × 2-Bit Multiplier	3	32	11.67	13

Results of these experiments are shown in table 2. The truth table for the 1-bit adder can be seen to be less regular (in terms of minimum gate complexity) than many of its close variants produced by single bit-flips. All 8 rows of the table have such "more regular" variants. This fits very well with the proposed explanation for evolutionary generalisation: the 1-bit adder violates the *regularity of arithmetic* hypothesis, and the 1-bit adder is the one function for which evolution failed to generalise in this study. For most of the "favoured errors" across all arithmetic functions, evolution was able to find a smaller circuit which implemented the "error" than one which implemented the arithmetically correct function. An example is shown in figure 2. Consequently, by Occam's razor[4] (the principle that one should favour the simplest explanation), these "errors" are actually better generalisations from a learning theory point of view than the original arithmetic functions.

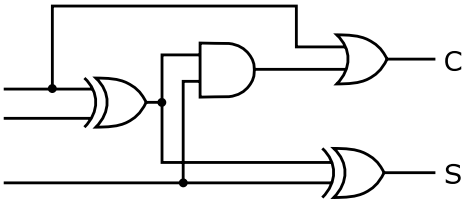


Fig. 2. A "more regular" variant of the 1-bit binary adder, where $1 + 0 + 0 = 3$ but all other rows are as per standard addition. The actual binary adder function requires 5 gates to implement

7.2 Random Search and Evolution

An experiment was run on the 1-bit adder to investigate all four possible variants of the 1-bit adder for the row $1 + 0 + 0$. 1000 evolutionary runs were performed for each variant truth table. A further 1000 (pseudo-)random search runs were performed with the row $1 + 0 + 0$ ablated: these runs used the same genetic encoding as the evolutionary runs, but the genome was completely randomised at every generation. Again, no explicit pressure for small circuits was present. Random search was terminated when a 100% correct circuit was found for the partial truth table with row $1 + 0 + 0$ ablated. The random search numbers are thus an estimate of how relatively frequent these variants actually are in the set of all possible encoded circuits. The results are shown in table 3.

Table 3. Comparison of 4 variants of the 1-bit adder for complexity, evolutionary difficulty, evolutionary "guessability" and estimated relative frequency in genome space (from random search experiment)

Variant	Min Evolved Size (Gates)	Mean Evolved Size (Gates)	Mean Evolution Time (Evals)	Frequency Evolved When Row Ablated	Estimated Frequency in Genome Space
$1+0+0=0$	6	12.1	7315	1%	0.7%
$1+0+0=1$	5	10.6	2892	18%	15.9%
$1+0+0=2$	6	11.3	6661	7%	2.3%
$1+0+0=3$	4	9.9	3135	74%	81.1%

We can see that $1+0+0 = 3$ is most frequent in the genome space, followed by $1+0+0 = 1$, $1+0+0 = 2$ and finally $1+0+0 = 0$. This ordering corresponds to the ordering by minimum gate complexity (except that the final two are tied for minimum gate complexity). We also see that evolution is reasonably effective at finding circuits for rare functions when the entire truth table is used to determine the fitness function. However, when the row is ablated, evolved solutions tend to approximately represent the underlying distribution of functions in genome space (supporting the *lack of evolutionary bias* assumption from section 6.3).

8 Conclusion

This paper contains empirical results which establish that evolved digital logic circuits can indeed do better than chance guessing when presented with previously unseen inputs from binary arithmetic problems. I have argued that this makes sense from the point of view of regularity: it is simply easier to implement a regular truth table (one with a smaller gate complexity) than an irregular one. The one-bit adder function was an exception to this rule, and experiments indicated that this truth table can reasonably be considered irregular compared to its near variants. Previous researchers have missed this phenomenon due to their focus on functionally perfect circuits and, in one case, an erroneous assumption about the output distribution of combinational circuits.

References

1. L. Barnett. Tangled webs: Evolutionary dynamics on fitness landscapes with neutrality. Master's thesis, Brighton, 1997.
2. K. Imamura, J. A. Foster, and A. W. Krings. The test vector problem and limitations to evolving digital circuits. In *Proceedings of the 2nd NASA/DoD Workshop on Evolvable Hardware*, pages 75–80. IEEE Press, 2000.
3. J. R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs.* MIT Press, 1994.
4. M. Li and P. Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, second edition, 1997.
5. M. A. Lones. *Enzyme Genetic Programming.* PhD thesis, Department of Electronics, University of York, 2003.
6. S. McGregor. Evolutionary and adaptive systems master's dissertation 2004. Master's thesis, University of Sussex, 2004.
7. J. Miller. What bloat? cartesian genetic programming on boolean problems. In *Late Breaking Papers, Proceedings of the 3rd Genetic and Evolutionary Computation Conference (GECCO'01)*, pages 295 –302, 2001.
8. J. F. Miller, T. Kalganova, N. Lipnitskaya, and D. Job. The genetic algorithm as a discovery engine: Strange circuits and new principles. In *Proceedings of the AISB Symposium on Creative Evolutionary Systems (CES'99)*, 1999.
9. J. F. Miller and P. Thomson. Aspects of digital evolution: Geometry and learning. In M. Sipper, D. Mange, and A. Perez-Uribe, editors, *Proceedings of Second International Conference on Evolvable Systems (ICES'98)*, pages 25–35. Springer-Verlag, 1998.
10. M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, 1998.
11. T. Naemura, T. Hashiyama, and S. Okuma. Module generation for genetic programming and its incremental evolution. In C. Newton, editor, *Second Asia-Pacific Conference on Simulated Evolution and Learning*, Australian Defence Force Academy, Canberra, Australia, 24-27 1998.

How Niche Construction Can Guide Coevolution

Reiji Suzuki and Takaya Arita

Graduate School of Information Science, Nagoya University,
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
{reiji, arita}@nagoya-u.jp
http://www2.create.human.nagoya-u.ac.jp/~{reiji, ari}/

Abstract. Niche construction is the process whereby organisms, through their metabolism, activities, and choices, modify their own and/or each other's niches. Our purpose is to clarify the interactions between evolution and niche construction by focusing on non-linear interactions between genetic and environmental factors shared by interacting species. We constructed a new fitness landscape model termed the NKES model by introducing the environmental factors and their interactions with the genetic factors into Kauffman's NKCS model. The evolutionary experiments were conducted using hill-climbing and niche-constructing processes on this landscape. Results have shown that the average fitness among species strongly depends on the ruggedness of the fitness landscape (K) and the degree of the effect of niche construction on genetic factors (E). Especially, we observed two different roles of niche construction: moderate perturbations on hill-climbing processes on the rugged landscapes, and the strong constraint which yields the convergence to a stable state.

1 Introduction

All living creatures partly modify their own and/or each other's niches as sources of selection through their metabolism, their activities, and their choices. This process is calld "niche construction" [1], and there are many evidences that it has strong effects on the evolution of organisms although it had been neglected for a long time in evolutionary biology.

A typical example of a niche-constructing organism is earthworms that change the structure and chemistry of soils through their burrowing activities. These changes are accumulated over generations, and then bring about different environmental conditions which expose successive population to different selection pressure. This effect is also called "ecological inheritance", which makes the generation inherit both genes and a legacy of modified selection pressures from ancestral organisms. In addition, these changes can affect the other organisms' evolution in the soil. The niche-constructing processes are observed in various taxonomic groups such as bacteria (decomposition of vegetative and animal matter), plants (production of oxygen), non-human animals (nest building) and humans (cultural process) [2].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 373–382, 2005.

The theoretical investigations into the effects of niche construction on evolution have been based mainly on the population genetics. For instance, Laland *et al.* constructed two-locus models, in which one locus affects the niche-constructing behavior which produces the resources in the environments and the fitness of the other locus is affected by the amount of accumulated resources [3]. They also introduced the ecological inheritance into their models in which the current amount of resources not only depends on the niche construction of the current individuals but also depends on the results of niche construction in previous generations. The results showed that niche construction and ecological inheritance yield unexpected results such as the maintenance of polymorphisms and the evolutionary momentum. The niche construction is now getting much attention in the field of artificial life. Taylor presented an individual-based model of niche construction [4]. In his model, the fitness of each individual is determined by other neighbors' gene expressions in its local environment. The results showed that the complex changes in the environmental states by the niche-constructing traits caused an evolution of organism with more genes which implies a continuous increase in the complexity of organisms. These studies describe the basic dynamics of the effects of niche construction within one species.

It is also essential to clarify the effect of the niche construction in the context of the coevolution of multiple species. There are a lot of evidences that the niche-constructing process of one species affected the course of evolution of the other species due to the modifications of the shared environment [2]. For example, many species of birds have evolved to use spider-web silk in their nest construction. Some species of snakes have evolved the behavior of waiting by the trails made by mammalian prey to ambush them. The burrowing behavior of earthworms that we have explained above also provides sites of microbial activity and soil environments for plant species by mixing organic matter in the soils. However, there have been few theoretical or constructive approaches which focused on the universal nature of the coevolutionary dynamics among species under the assumption of the indirect interactions via niche construction and ecological inheritance, although the indirect genetic effects within one species have been discussed [5].

Recently, Hui *et al.* introduced a niche-constructing trait into a lattice model of the evolution of metapopulation [6]. They assumed that there were superior-inferior relationship between several species and the inferior species only conducted niche construction that produces the resources which affect their survival. The result showed that the strength of the effect of interspecific niche construction strongly affects the coexistence of species, and the segregation of species' distribution. However, the results were quite specific to a priori setting of relationships among species and the evolution of the niche-constructing trait itself was not introduced. Thus the general dynamics of indirect interactions among different species sharing the same environment is still unclear.

Our purpose is to clarify the complex relationships between evolution and niche construction by focusing on non-linear interactions between genetic and environmental factors shared by interacting species [8]. For this purpose, we

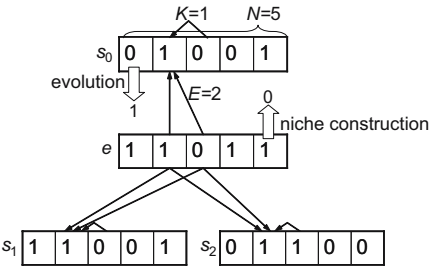


Fig. 1. An example of the NKES model when N=5, K=1, E=2 and S=3

have constructed a new fitness landscape model termed the NKES model by introducing the environmental factors and their interactions with the genetic factors into Kauffman's NKCS model [7]. Then, we conducted the evolutionary experiments based on the hill-climbing and the niche-constructing processes of species on this landscape, in which each species can increase its own fitness by changing not only its genetic factors but also the environmental factors. With experiments using various settings of the ruggedness of fitness landscape and the strength of the effect of niche construction on the fitness of genetic factors, we clarify how niche-constructing behaviors can facilitate the adaptive evolution of interacting species via the shared environment.

2 Model

2.1 NKES Fitness Landscape

We constructed the NKES model by introducing environmental factors and their interactions with the genetic factors into Kauffman's NKCS model [7]. There are S species who share the same environment of which properties are described as N-length binary values e_i (i=0, ..., N-1). We define e_i as environmental factors which represent abstract conditions of the shared environment such as the chemistry of soil, the temperature, the humidity, the existence of burrows, nests and resources. Each species s_i (i=0, ..., S-1) has N genetic factors represented as binary values $g_{i,j}$ (j=0, ..., N-1).

The fitness of each genetic factor $g_{i,j}$ has epistatic interactions not only with other K genetic factors $g_{i,(j+k) \bmod N}$ (k=1, ..., K) in its own species but also has non-linear interactions with E environmental factors $e_{(j+l) \bmod N}$ (l=0, ..., E-1). The fitness contribution of each genetic factor caused by interactions among genetic and environmental factors is defined in similar manner to the NKCS model. For each $g_{i,j}$, we prepare a lookup table which defines its fitness corresponding to all possible (2^{K+E+1}) combinations of interacting genetic and environmental factors. The value of each fitness in the lookup table is randomly set within the range of $[0.0, 1.0]$. The fitness of each species is regarded as the average fitness over all of its genetic factors. Thus, the parameter K represents the ruggedness

of the fitness landscape of each species and E represents the strength of the effect of niche construction on the fitness of genetic factors in this model. Figure 1 shows an example image of this model when N=5, K=1, E=2 and S=3. Each table represents a set of values of genetic or environmental factors, and thin arrows that issue from these values represent the existence of non-linear effects on values of other genetic or environmental factors.

2.2 Evolution and Niche Construction

In each generation, each species independently chooses the process which yields the best increase in its own fitness from "evolution", "niche construction" or "doing nothing" by using the following procedures: First, we calculate the fitness of the species when a randomly-selected genetic factor is flipped. At the same time, we also calculate its fitness when a randomly-selected environmental factor is flipped. The former value corresponds to the possible result caused by the evolutionary process. The latter corresponds to the possible result by the niche-constructing process, that is, the evolution of the niche-constructing trait which modifies the corresponding environmental factor. Then, the species adopts the process which brings about the best fitness by comparing these two fitness and its current fitness. If the current fitness is the best, it does nothing in this generation. After all species have chosen the processes, they actually conduct the adopted processes at the same time. Note that if more than one species decide to flip the same environmental factor, it is flipped only once in each generation.

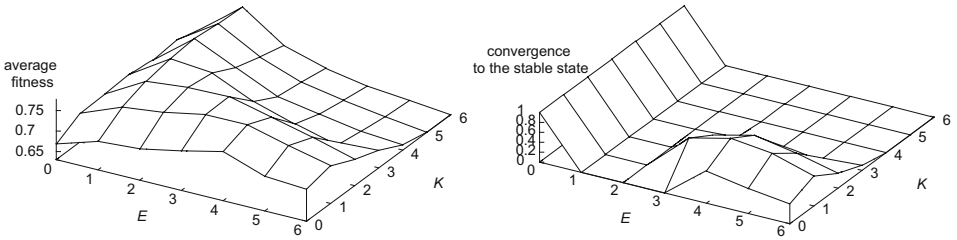
The outlined arrows in Figure 1 represent examples of evolutionary process and niche-constructing process. If one species flips the environmental factor by niche construction, it can change the fitness contributions of the other species' genetic factors, and then can bring about different evolutionary or niche-constructing dynamics of the other species. There are indirect interactions among species via niche constructions instead of the direct interactions among them like the NKCS model.

3 Experimental Results

3.1 General Analyses

We have conducted experiments using various settings of K and E (N=80 and S=3) for 100000 generations. The initial values of genetic and environmental factors were randomly decided. Firstly, we focus on the effects of K and E on the average fitness among all species. The average fitness does not only represent how the species could evolve on the current environment but also shows how the environment was modified and became better for all species through niche constructions.

Figure 2 (a) shows the average fitness among all species during the last 1000 generations in various cases of K and E. The x and y axes correspond to the conditions of K and E, and the z axis represents the average fitness on



(a) average fitness (b) convergence to the stable state

Fig. 2. The average fitness and the proportion of the convergence to stable state in various cases of K and E

corresponding conditions. Each value is the average over 20 trials. The first thing we notice is that the average fitness is large (exceeds 0.75) when either K or E is relatively small. In particular, there are two different conditions which created the peaks of the average fitness: the cases when K=4 and E=1 (0.78), and when K=1 and E=4 (0.77). Figure 2 (b) also shows the proportion of trials in which the population completely converged to a stable state, in other words, the fitness of any species can not be improved by neither evolution nor niche construction. There is a peak of the proportion of convergence (0.95) in the latter condition, while it is 0.0 in the former condition. It implies that different dynamics of evolution and niche construction brought about the high average fitness under both conditions.

3.2 Evolutionary Dynamics When K=4 and E=1

Here, we investigate in detail the two conditions which brought about the high fitness respectively. First, we focus on the case when K=4 and E=1. In this case, it should be noticed that the average fitness was higher than the corresponding condition without niche construction (K=4 and E=0). When E=0, each species is able to climb the fitness landscape to increase its fitness only by changing its genetic factors, and rapidly gets stuck in the local optimum. Actually, Figure 2 (b) shows that the population always converged to a stable state in all cases of E=0.

However, when E=1, each species can change its fitness landscape by the niche-constructing process. Figure 3 shows a typical transition of the average fitness among species during the first 30000 generations. Note that the transition of the fitness of each species was approximately similar to that of the average fitness but with modest fluctuation. We can see that the species smoothly increased their fitness and fluctuated around 0.78, but they never converged to a stable state.

In this model, the niche construction does not only simply increase the fitness of the performer of the niche construction, but also can change the other species' fitness by changing their fitness landscapes. The difference in the average fitness between with and without niche construction is mainly caused by the latter effect

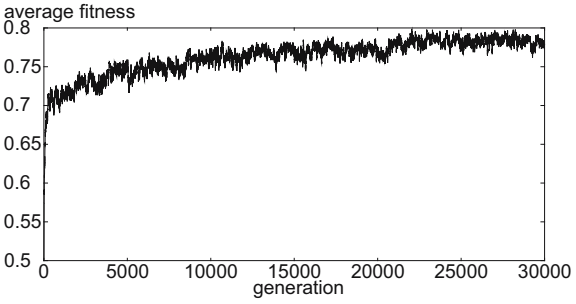


Fig. 3. The transition of the average fitness when K=4 and E=1

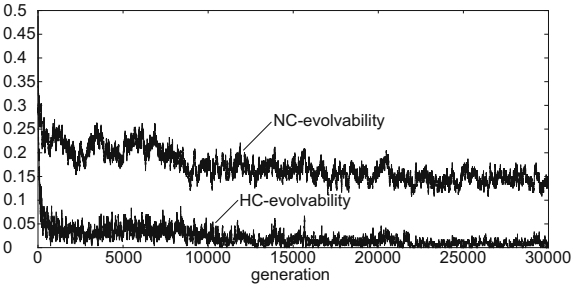


Fig. 4. The transitions of the HC-evolvability (thin line) and NC-evolvability (thick line) when K=4 and E=1

of niche construction. Figure 4 shows the transition of the average evolvability provided by hill climbing (HC-evolvability) and the average evolvability provided by niche construction (NC-evolvability) among species in the same experiment as the one shown in Figure 3. The HC-evolvability (or NC-evolvability) represents the average proportion of genetic (or environmental) factors for each species which can increase its own fitness by flipping them. These indices measure how often each species can apply the evolutionary or niche-constructing process in order to increase its fitness. Figure 4 shows that the NC-evolvability kept a relatively large value, while the HC-evolvability approached to almost 0.0 after the drastic decrease in both indices until a few hundreds generation. This means that the species were almost getting to local optimums, but the continuous niche constructions through generations prevented them from getting stuck in the local optimums by slightly changing their landscapes and enabled them to obtain higher fitness regardless of their high ruggedness. Thus, the niche construction worked as a moderate perturbation on the other species' hill-climbing processes in this case.

3.3 Evolutionary Dynamics When K=1 and E=4

The other condition which yielded the high average fitness is the case of K=1 and E=4. The important difference compared with the previous condition is

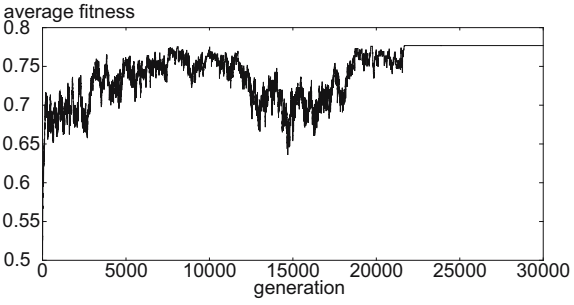


Fig. 5. The transition of the average fitness when K=1 and E=4

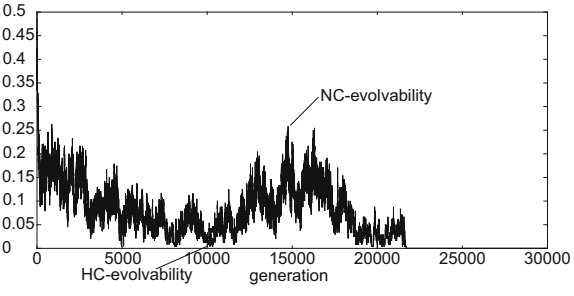


Fig. 6. The transitions of the HC-evolvability (thin line) and NC-evolvability (thick line) when K=1 and E=4

that the population converged to a stable state in almost all trials as shown in Figure 2 (b). Figure 5 and 6 show the typical transitions of indices respectively. We observe the average fitness completely converged to 0.78 around 22000th generation after its temporal increase and subsequent decrease from the initial population. Such a temporal decrease is interesting as all species are always trying to increase their own fitness in our model. Also, the transitions of two evolvability in Figure 6 were quite similar though the HC-evolvability was just slightly smaller than the NC-evolvability.

These phenomena are supposed to occur due to the following reason: As shown in Figure 2 (b), the high average fitness was caused by the convergence to a stable state in this case. It means that the HC-evolvability and NC-evolvability became 0.0 at the same time as shown in Figure 6. Figure 7 shows the transitions of the NC-evolvability when K=1, and E=1, 2 and 6. We see that the NC-evolvability tends to approach to the smaller value in case of E=2 in comparison with the case of E=1. It is because that the strong effects of niche construction on the fitness of genetic factors make the species difficult to improve its fitness by niche construction likewise the species more easily gets stuck in the local optimum on the standard NK fitness landscape as K increases. However, the increase in also E brings about the large fluctuation around the relatively large value in NC-evolvability if E is too large such as the case of E=6 in Figure 7.

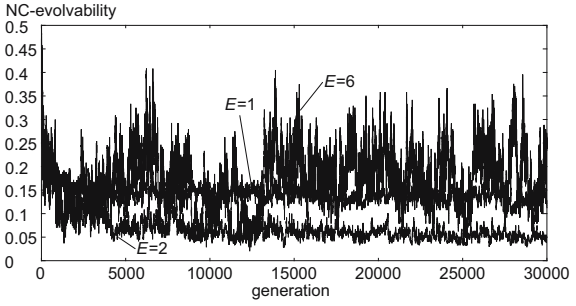


Fig. 7. The transitions of NC-evolvability when K=1, and E=1, 2 and 6

It is because that as E becomes large, the change in the environmental factor by niche construction of one species more drastically changes the other species' fitness landscapes and draws them back into the bottom of their landscapes. Thus, NC-evolvability frequently approaches to 0.0 when these effects are well-balanced (Figure 6).

Also, as E increases, the transition of HC-evolvability tends to be synchronized with the NC-evolvability as shown in Figure 6, and the fluctuation in HC and NC-evolvability becomes larger as K increases (not shown) because the increase in K makes the species more frequently conduct the niche constructions. Thus, the convergent state occurs the most frequently only when K is small and E is large.

4 Comparison with the NKCS Model

Kauffman proposed the NKCS model as a mathematical model designed to investigate the evolutionary dynamics of evolving species in which each species' genetic factor directly affects other species' fitness [7]. In his model, each genetic factor has epistatic interactions not only with other K genetic factors in its own species but also has interactions with C genetic factors in other S species respectively. It is well known as a good abstract model of directly coevolving species. Here, we compare the evolutionary dynamics of the NKES model with that of the NKCS model so as to clarify how the difference in the structures of (direct or indirect) interactions among species changes the coevolutionary process.

The important fact is that the evolutionary dynamics in the NKES model are quite complex compared with that in the NKCS model. Table 1 shows how HC and NC-evolvability are affected by the increase in the parameters K, C and E in the NKCS or NKES model. In the NKCS model, it is well known that the population tends to rapidly converge to a stable state (ESS condition) when K is relatively large and C is relatively small if the evolutionary processes are conducted in similar manner to that in the NKES model [7]. The typical effects of K and C in the NKCS model become quite simple as shown in Table 1.

In contrast, the effects of K and E in the NKES model are not simple because we have to consider their complex effects on both HC and NC-evolvability. The

Table 1. The effect of increase in K, C and E on HC/NC-evolvability in NKCS and NKES model

	increase in K	increase in C or E
NKCS	decrease in HC-evolvability	increase in the other species' HC-evolvability
NKES	(1) decrease in HC-evolvability (2) increase in the other species' NC-evolvability	(1) decrease in NC-evolvability (2) increase in the other species' NC-evolvability

similar effect of K on the HC-evolvability in the NKCS model also exists in the NKES model. But it brings about the increase in the other species' NC-evolvability because the species frequently conduct niche-constructing behaviors if it is difficult for the species to improve its fitness by evolutionary process. As a result, the population tends to become unstable as K increases. The increase in E also has two different effects on the NC-evolvability as discussed in the previous section, and it make the population stable on the condition that the both effects are well balanced.

As a whole, it should be noticed that the population tends to become completely stable on the opposite condition (K is small and E is large) in the NKES model compared with the condition on which the population rapidly converges to a stable state in the NKCS model (K is large and C is small).

5 Conclusion

We have discussed the universal nature of interactions between evolution and niche construction by using the NKES fitness landscape model. We found that the average fitness among species strongly depends on the ruggedness of fitness landscape (K) and the strength of the effect of niche construction on the genetic factors (E). It should be emphasized that the two qualitatively different roles of niche construction brought about the high average fitness in different conditions. When K is large and E is small, the niche construction by one species works as moderate perturbations on the other species' hill-climbing processes on the highly rugged landscapes, which prevents them from getting stuck in the local optimums. On the other hand, when K is small and E is moderately large, the strong effect of niche constructions on the fitness of genetic factors yields the convergence to a completely stable state which maintain the high average fitness.

There are some examples of the stable and symbiotic relationships among interacting species which mutually modify the shared environment. Some ants have a mutualism with acacia trees [2]. The ants destroy seedlings and attacks mammalian browsers and insect pests. In turn, the acacia provides thorns and nectarines that house and feed the ants. It is also well known that flowering plants have evolved to provide nectar for several insects. The insects attracted by the flowers facilitate pollination process of these plants by providing the movement

of pollen while gathering nectar in return. Mutual modification through niche construction is essential for the fitness of both species, which establishes tightly-coupled relationships between the niche-constructing species and environmental factors. In this sense, these types of ecosystems might be explained as the latter case of our experiments (K is small and E is large) rather than the former. We believe that the evolution observed in this case reflects some aspects of the establishment of these tightly-coupled relationships in real biological systems.

Future work includes investigations into the effects of the other parameters on the roles of niche construction and the introduction of the evolution of the network structure among species and environments.

References

1. Odling-Smee, F. J.: Niche Constructing Phenotypes, Plotkin, H. C. (ed), *The Role of Behavior in Evolution*, pp. 73–132, MIT Press (1988).
2. Odling-Smee, F. J., Laland, K. N. and Feldman, M. W.: *Niche Construction -The Neglected Process in Evolution-*, Princeton University Press (2003).
3. Laland, K. N., Odling-Smee, F. J. and Feldman, M. W.: Evolutionary Consequences of Niche Construction: A Theoretical Investigation Using Two-locus Theory, *Journal of Evolutionary Biology*, 9: 293–316 (1996).
4. Taylor, T.: Niche Construction and the Evolution of Complexity, *Proceedings of Artificial Life IX*, pp. 375–380 (2004).
5. Wolf, J. B., Brodie III, E. D., Cheverud, A. J., Moore, A. J. and Wade, M. J., Evolutionary Consequences of Indirect Genetic Effects, *Trends in Ecology and Evolution*, 13: 64–69 (1998).
6. Hui, C., Li, Z. and Yue, D.: Metapopulation Dynamics and Distribution, and Environmental Heterogeneity Induced by Niche Construction, *Ecological Modeling*, 177: 107-118 (2004).
7. Kauffman, S.: *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford University Press (1993).
8. Suzuki, R. and Arita, T.: Evolution and Niche Construction in NKES Fitness Landscape, *Proceedings of the 10th International Symposium on Artificial Life and Robotics*, pp. 493–496 (2005).

Measuring Diversity in Populations Employing Cultural Learning in Dynamic Environments

Dara Curran and Colm O'Riordan

National University of Ireland, Galway, Ireland
dara.curran@nuigalway.ie
colmor@it.nuigalway.ie

Abstract. This paper examines the effect of cultural learning on a population of neural networks. We compare the genotypic and phenotypic diversity of populations employing only population learning and of populations using both population and cultural learning in two types of dynamic environment: one where a single change occurs and one where changes are more frequent. We show that cultural learning is capable of achieving higher fitness levels and maintains a higher level of genotypic and phenotypic diversity.

1 Introduction

A number of learning models may be readily observed from nature and have been the focus of much study in artificial intelligence research. Population learning (i.e. learning which occurs at a population level through genetic material) is typically simulated using genetic algorithms. Life-time learning (i.e. learning which takes place during an organisms's life time through reactions with its environment) can be simulated in a variety of ways, typically employing neural networks or reinforcement learning models.

A relatively new field of study in artificial intelligence is synthetic ethology. The field is based on the premise that language and culture are too complex to be readily analysed in nature and that insight can be gained by simulating its emergence in populations of artificial organisms. While many studies have shown that lexical, syntactical and grammatical structures may spontaneously emerge from populations of artificial organisms, few discuss the impact such structures have on the relative fitness of individuals and of the entire population.

A robust multi–agent system should be able to withstand and adapt to environmental changes. This type of behaviour parallels that of the natural world where species capable of adaptation will have more chance of evolutionary success than ones that are rigid and incapable of such plasticity. At its most basic level, adaptation in nature takes the form of population learning. At a higher level, organisms capable of adapting their behaviour to suit a particular environment during their lifetimes will be more likely to survive in the long term.

The focus of this paper is to attempt to understand the effect of cultural learning on a population of artificial organisms subjected to dynamic environments. This is accomplished by studying its effect on the population's fitness

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 383–392, 2005.

as well as its genotypic and phenotypic diversity. The remainder of this paper is arranged as follows. Section 2 introduces background research, including descriptions of diversity measures and cultural learning techniques that have been employed for this study. Section 3 describes the experimental setup. Section 4 presents the Experiment Results and Section 5 presents conclusions.

2 Background Research

2.1 Cultural Learning

Culture can be succinctly described as a process of information transfer within a population that occurs without the use of genetic material. Culture can take many forms such as language, signals or artifactual materials. Such information exchange occurs during the lifetime of individuals in a population and can greatly enhance the behaviour of such species. Because these exchanges occur during an individual's lifetime, cultural learning can be considered a subset of lifetime learning.

An approach known as synthetic ethology [10,17] argues that the study of language is too difficult to perform in real world situations and that more meaningful results could be produced by modeling organisms and their environment in an artificial manner. Artificial intelligence systems can create tightly controlled environments where the behaviour of artificial organisms can be readily observed and modified. Using genetic algorithms, the evolutionary approach inspired by Darwinian evolution, and the computing capacity of neural networks, artificial intelligence researchers have been able to achieve very interesting results.

In particular, experiments conducted by Hutchins and Hazlehurst [8] simulate cultural evolution through the use of a hidden layer within an individual neural network in the population. This in effect, simulates the presence of a Language Acquisition Device (LAD), the physiological component of the brain necessary for language development, the existence of which was first suggested by Chomsky [3]. The hidden layer acts as a verbal input/output layer and performs the task of feature extraction used to distinguish different physical inputs. It is responsible for both the perception and production of signals for the agent.

A number of approaches were considered for the implementation of cultural learning including fixed lexicons [19], indexed memory [16], cultural artifacts [7] and signal–situation tables [10]. The approach chosen was the teacher/ pupil scenario [4,2] where a number of highly fit agents are selected from the population to act as teachers for the next generation of agents, labelled pupils. Pupils learn from teachers by observing the teacher's verbal output and attempting to mimic it using their own verbal apparatus. As a result of these interactions, a lexicon of symbols evolves to describe situations within the population's environment.

2.2 Diversity

Diversity measures typically quantify the differences between individuals in a population. It is commonly accepted that a population that is capable of maintaining diversity will avoid premature convergence and local maxima.

Diversity measures for populations of neural networks have been the focus of considerable research, focusing mainly on genotypic diversity [18,14,1]. Many methods exist for the calculation of genotypic diversity, many based on binary representations. For the purposes of this research however, many schemes are unsuitable due to the nature of the marker-based encoding scheme used to represent each neural network.

Our scheme examines each block of the encoding and compares it to blocks of similar length in other encodings. Each encoding block contains a single node and a number of links emanating from that node. It is therefore intuitive to propose that blocks of similar length (having a similar number of emanating links) are suitable for mutual comparison.

There is comparatively little research on phenotypic diversity in evolutionary computation. Typically, phenotypic diversity is measured at the fitness level [5]. However, this measure tends to compress the available diversity information resulting in a coarse grained measure not useful in all situations. The approach adopted in this work is to examine the components of the fitness value of each individual, i.e. an individual's response to each bit-parity stimulus. By comparing the difference between all responses (and not just the aggregate fitness function) a finer grained measure of phenotypic diversity can be obtained.

2.3 Dynamic Environments

Many approaches have been taken to simulate changing environments for multi–agent and artificial life systems[13,6,15,11] focusing on Latent Energy Environments and fitness functions which vary over time. Our approach, while straightforward, has the advantage of clarity: agents are repeatedly presented with a number of bit–patterns representing either food or poison. An agent capable of distinguishing the two by correctly ingesting food and avoiding poison will be rewarded with a high fitness level and reproductive opportunity. At each environmental change all bit–patterns representing food are made to represent poison and vice–versa thus completely reversing the environment. This is partly based on work performed by Nolfi et al[13] who compared the performance of a robotic agent employing genetic evolution (population learning) and that of agents employing back–propagation (life–time learning) in a changing environment.

3 Simulator

The architecture of the artificial life simulator can be seen as a hierarchical structure. At the top-level of the simulator is a command interpreter which allows users to define an experiment's variables including the number of networks, the number of generations to run the experiment, mutation and crossover rates and the actual problem set which the population will be attempting to solve.

The neural network layer takes the variables set using the command interpreter and initialises a given number of neural networks. The layer then performs training and testing of the networks according to the parameters of the experiment. These network memory structures are then passed to the encoding layer

which transforms them into genetic code structures for use in the genetic algorithm. The encoding mechanism used for this set of experiments is a modified version of marker based encoding.

Marker based encoding represents neural network elements (nodes and links) in a binary string. Each element is separated by a marker to allow the decoding mechanism to distinguish between the different types of element and therefore deduce interconnections[9,12].

In this implementation, a marker is given for every node in a network. Following the node marker, the node's details are stored in sequential order on the bit string. This includes the node's label and its threshold value. Immediately following the node's details, is another marker which indicates the start of one or more node–weight pairs. Each of these pairs indicates a back connection from the node to other nodes in the network together with connection's weight value. Once the last connection has been encoded, the scheme places an end marker to indicate the end of the node's encoding

The genetic algorithm layer uses the genetic codes and the data retrieved from the neural network layer's testing of the networks to perform its genetic operators on the population. A new population is produced in the form of genetic codes. These are passed to the decoding layer which transforms each code into a new neural network structure. These structures are then passed up to the neural network layer for a new experiment iteration. Once the required number of generations has been reached, the experiment finishes.

Two-point crossover is employed and weight mutation is employed which takes the weight value and increases/decreases the value according to a random percentage (200%). This approach was found, empirically, to be more successful and was adopted for this set of experiments.

3.1 Simulating Cultural Evolution

In order to perform experiments related to cultural evolution, it was necessary to adapt the existing simulator architecture to allow agents to communicate with one another. This was implemented using an extended version of the approach adopted by Hutchins and Hazlehurst. The last hidden layer of each agent's neural network functions as a verbal input/output layer.

At end of each generation, a percentage of the population's fittest networks are selected and are allowed to become teachers for the next generation. The teaching process takes place as follows: a teacher is stochastically assigned n pupils from the population where $n = \frac{N_{pop}}{N_{teachers}}$, where N_{pop} is the population size and $N_{teachers}$ is the number of teachers. Each pupil follows the teacher in its environment and observes the teacher's verbal output as it interacts with its environment. A teaching cycle occurs when the pupil attempts to emulate its teacher's verbal output using back-propagation. Once the number of required teaching cycles is completed, the teacher networks die and new teachers are selected from the new generation.

Unlike previous implementations, the number of verbal input/output nodes is not fixed and is allowed to evolve with the population, making the system

more adaptable to potential changes in environment. In addition, this method does not make any assumptions as to the number of verbal nodes (and thus the complexity of the emerging lexicon) that is required to effectively communicate. It should be noted that neither the parent's nor the pupil's genotype is altered at any time during these cultural exchanges.

4 Experimental Setup

The following set of experiments each employs two populations. One population is allowed to evolve through population learning (by genetic algorithm), while the other employs both population and cultural learning. The problem domain for this set of experiments is the 5-bit parity problem. Each network is exposed to bit patterns and must determine whether the pattern represents an odd or even number. Fitness is assigned according to the mean square error of a network.

Two types of environment were employed for the experiments: an environment with a single dramatic change (at generation 200) and another with a series of regular changes (every 20 generations) during the course of the experiment. The change in environment is implemented by reversing the food and poison representations such that the bit pattern representing food will represents poison and vice-versa.

Each experiment consists of a population of 50 neural networks evolving for 400 generations with crossover and mutation rates set at 0.6 and 0.02 respectively. The population employing cultural learning takes the fittest 10% of each generation as teachers which interact with pupils for five teaching cycles. An additional parameter, cultural mutation, adds noise to each interaction with probability 0.02. The results presented are averaged from 10 independent runs.

5 Experiment Results

The experimental results are divided into two sections. The first examines the relative performance of cultural learning and population learning through analysis of the error values for each population. The second section is concerned with genotypic and phenotypic diversity measures for each population.

5.1 Single Environment Change

The average error values for both populations for the single environment change experiment are presented in figure 1. It is clear from the results that the population employing cultural learning is capable of reducing its error values more successfully than the population using population learning alone. The environment change at generation 200 is clearly marked by a large surge in error values occurring in both populations. However, the sharp increase in error is more evident in the population employing population learning alone, suggesting that cultural learning is softening the environment change.

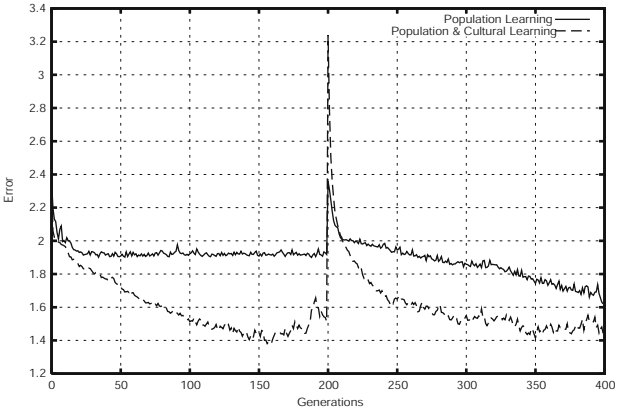


Fig. 1. Average Error

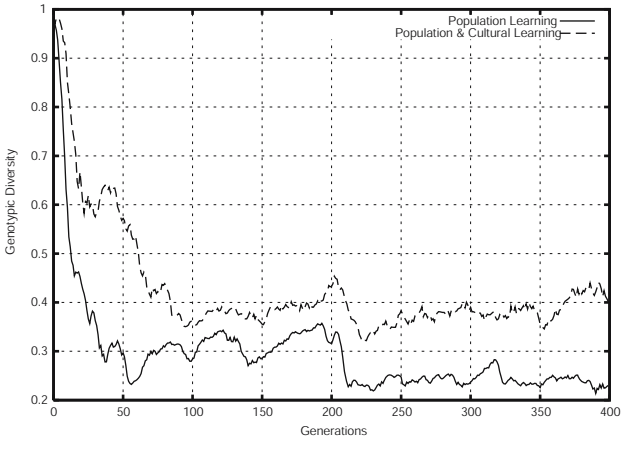


Fig. 2. Genotypic Diversity Values

Figure 2 shows the genotypic diversity for both populations. While both populations have a tendency to reduce diversity as the experiment progresses, the population employing cultural learning is capable of maintaining a higher (and statistically significant) level of diversity throughout the experiment. This trend is reinforced by the results of the phenotypic diversity measure, presented in figure 3. The phenotypic diversity of the population employing population learning alone is considerably lower than that of the population employing cultural learning.

It is clear from these results that in the single change environment, cultural learning is capable of maintaining a high genotypic and phenotypic diversity for its population. This can be correlated to its corresponding superior performance with regard to average error values.

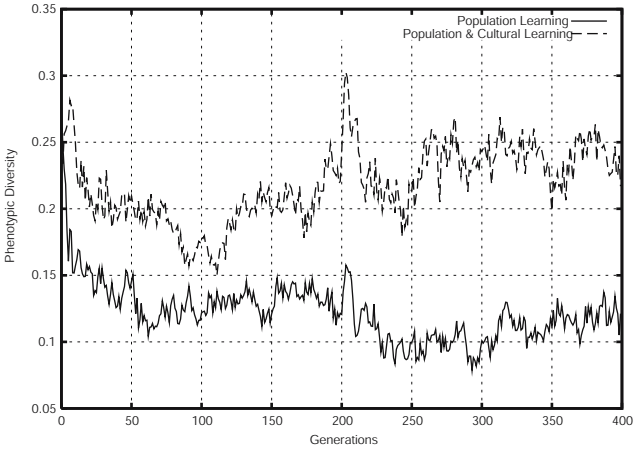


Fig. 3. Phenotypic Diversity Values

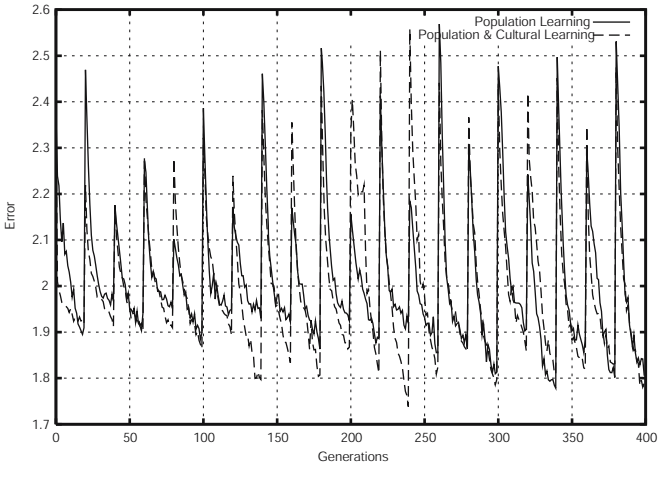


Fig. 4. Average Error

5.2 Multiple Environment Changes

The multiple environment change experiment presents a considerably more difficult challenge to both populations as the reversals in environment occur very frequently. Figure 4 presents the average error of both populations over the experiment run. Each environment change can be clearly seen as a surge in average error every 20 generations. Clearly both populations experience difficulty in tracking the environmental changes in this experiment.

The population employing cultural learning is capable of matching and in some cases improving on the error values acheived by the population employ-

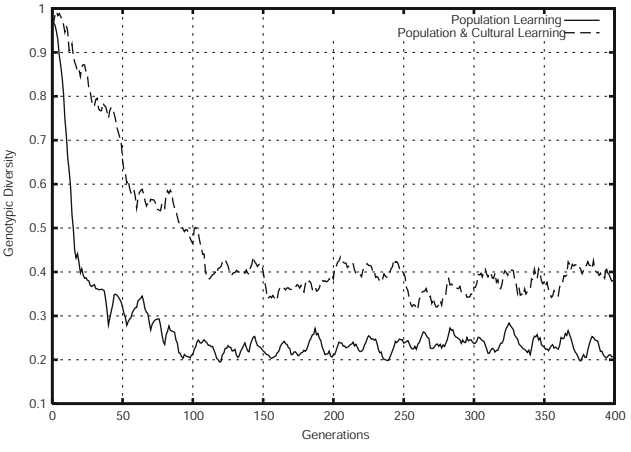


Fig. 5. Genotypic Diversity Values

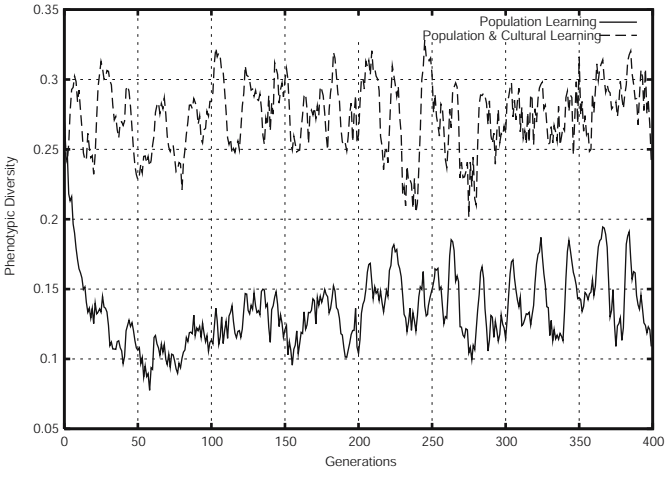


Fig. 6. Phenotypic Diversity Values

ing population learning alone. However, it cannot be said that there is a clear distinction between the two populations.

Figure 5 presents the results of the genotypic diversity measure for both populations. The results are similar to those obtained in the previous experiment set, with both populations reducing diversity over the experiment run, but with the population employing cultural learning maintaining a higher level throughout.

Similarly, the phenotypic diversity measure results outlined in figure 6 show that the population employing cultural learning is achieving and maintaining higher levels of phenotypic diversity than that of the population employing population learning alone.

6 Conclusions

The results presented in this paper suggest that the addition of cultural learning is beneficial to a population subjected to dramatic environmental changes, but is not capable of providing any real advantage in environments where changes occur more frequently. It should be stressed that we do not wish to generalise as to the effects of cultural learning for all problems, rather that this study provides a useful starting point into the analysis of the potential benefits of cultural learning. Diversity measures in particular may allow more detailed analysis into the effects of cultural learning for a variety of problem domains. Future work will focus on more complex problems and environments where changes occur more gradually, rather than simple reversal of problem solutions.

Acknowledgements

This research is funded by the Irish Research Council for Science, Engineering and Technology.

References

1. G. Brown. *Diversity in Neural Network Ensembles.* PhD thesis, University of Birmingham, 2003.
2. A. Cangelosi and D. Parisi. The emergence of a language in an evolving population of neural networks. *Technical Report NSAL–96004, National Research Council, Rome*, 1996.
3. N. Chomsky. On the nature of language. In *Origins and evolution of language and speech*, pages 46–57. Annals of the New York Academy of Science, New York. Vol 280, 1976.
4. D. Denaro and D. Parisi. Cultural evolution in a population of neural networks. In *M.Marinaro and R.Tagliaferri (eds), Neural Nets Wirn-96.New York: Springer*, pages 100–111, 1996.
5. G. Kendall E. K. Burke, S. Gustafson. Diversity in genetic programming: an analysis of measures and correlation with fitness. In *IEEE Trans. Evolutionary Computation*, volume 8(1), pages 47–62, 2004.
6. J. J. Grefenstette. Genetic algorithms for dynamic environments. In R. Maenner and B. Manderick, editors, *Parallel Problem Solving from Nature 2*, pages 137–144, 1992.
7. E. Hutchins and B. Hazlehurst. Learning in the cultural process. In *Artificial Life II, ed. C. Langton et al.*, pages 689–706. MIT Press, 1991.
8. E. Hutchins and B. Hazlehurst. How to invent a lexicon: The development of shared symbols in interaction. In N. Gilbert and R. Conte, editors, *Artificial Societies: The Computer Simulation of Social Life*, pages 157–189. UCL Press: London, 1995.
9. H. Kitano. Designing neural networks using genetic algorithm with graph generation system. In *Complex Systems, 4, 461-476*, 1990.
10. B. MacLennan and G. Burghardt. Synthetic ethology and the evolution of cooperative communication. In *Adaptive Behavior 2(2)*, pages 161–188, 1993.

11. F. Menczer. Changing latent energy environments: A case for the evolution of plasticity. In *Technical Report CS94-336*, 1994.
12. G. F. Miller, P. M. Todd, and S. U. Hedge. Designing neural networks using genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pages 379–384, 1989.
13. S. Nolfi and D. Parisi. Learning to adapt to changing environments in evolving neural networks. In *Technical Report 95-15, Institute of Psychology, National Research Council, Rome, Italy.*, 1995.
14. D. W. Opitz and J. W. Shavlik. Generating accurate and diverse members of a neural-network ensemble. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 535–541. The MIT Press, 1996.
15. T. Sasaki and M. Tokoro. Adaptation toward changing environments: Why darwinian in nature? In Phil Husbands and Inman Harvey, editors, *Fourth European Conference on Artificial Life*, pages 145–153, Cambridge, MA, 1997. MIT Press.
16. Lee Spector. Genetic programming and AI planning systems. In *Proceedings of Twelfth National Conference on Artificial Intelligence*, pages 1329–1334, Seattle, Washington, USA, 1994. AAAI Press/MIT Press.
17. L. Steels. The synthetic modeling of language origins. In *Evolution of Communication*, pages 1–34, 1997.
18. X. Yao Y. Liu and T. Higuchi. Evolutionary ensembles with negative correlation learning. In *IEEE Transactions on Evolutionary Computation*, volume 4(4), pages 380–387, 2000.
19. H. Yanco and L. Stein. An adaptive communication protocol for cooperating mobile robots. In *From Animals to Animats 2. Proceedings of the second International Conference on Simulation of Adaptive Behavior*, pages 478–485. MIT Press, Cambridge Ma., 1993.

On a Quantitative Measure for Modularity Based on Information Theory

Daniel Polani[1], Peter Dauscher[2], and Thomas Uthmann[2]

[1] University of Hertfordshire, Hatfield, UK
d.polani@herts.ac.uk
http://homepages.feis.herts.ac.uk/~comqdp1/
[2] Johannes Gutenberg Universität Mainz, Germany
{dauscher, uthmann}@informatik.uni-mainz.de
http://www.informatik.uni-mainz.de/~dauscher/

Abstract. The concept of modularity appears to be crucial for many questions in the field of Artificial Life research. However, there have not been many quantitative measures for modularity that are both general and viable. In this paper we introduce a measure for modularity based on information theory. Due to the generality of the information theory formalism, this measure can be applied to various problems and models; some connections to other formalisms are presented.

1 Introduction

In the studies of complex systems and Artificial Life, a central question is how it is possible that, over time, systems can emerge with ever increasing complexity. This question is particularly prominent if one considers the Darwinian evolution which, from a naive point of view, appears to be mainly directed random search with a large test population. However, even the powerful parallelism that is available to evolution in form of huge numbers of individuals cannot alone explain how the vast search space of possible configurations of living organisms can be efficiently searched and exploited towards increasing complexity.

It seems that nature employs to some degree the same method as human programmers in large software systems (or vice versa). In the latter, with the advent of the software crisis in the 1970s [1] , it became clear that large monolithic software systems in which each part depends on many others (a form of nonlocality) are unmanageable. Even if they should work reasonably reliably at a certain point in time, they cannot easily be adapted to new tasks. This is being solved by introducing *modules* which solve subproblems independently from the rest of the system and organizing larger systems by building them up from these smaller, manageable modules.

Adaptability is one of the central motifs of natural evolution. Therefore, the question arises whether evolution manages complexity in a similar way as human software engineers, via modularity. It turns out that there are several phenomena in nature that can be construed as exhibiting elements of modularity. The

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 393–402, 2005.

existence of genes that encode certain traits of the phenotype, the crossover operator (which is construed by researchers of artificial evolution as to be preserving *building blocks* which encode for separable (i.e. modular) properties of the phenotype. Of course, the situation in systems evolving in nature is much more involved as there is no human designer, but even in artificial software systems pure modularity does not exist in general.

As different as the different instances of systems are that exhibit or do not exhibit modularity and as different the language is that is being used in conjunction with those, they seem to share common properties. It would be very useful to formalize these properties in a common language. It would enable us to understand better what modularity is, when it can be made use of, or even when we can expect it to emerge [2,3,4] thus helping us to obtain further clues how natural evolution manages to climb the ladder of complexity.

It seems, however, that only relatively recently systematic approaches have been made to decomposition of tackle the decompositional structures of complex systems [5]. In the wake of the success of information-theoretic methods in the study of dynamical systems [6], recent approaches to address the question of modular decomposition of networks convert static networks into dynamic systems via a diffusion dynamics approach and analyse it applying spectral graph and information theory [7,8]. Independence graphs derived from probabilistic relations [9] and related information-theoretical notions [10] provide a growing toolbox to address these questions.

The specific systems we will address here already have an a priori dynamical structure and do not require it to be artificially imposed, as done in above models. At the same time, they are of high relevance for Artificial Life studies. We begin with the illustrative and inspiring model from [11] and show how information theory can be used in order to adapt its classifications as to become both more intuitive and finer-grained. We will then relate this approach to modularity arising due to the variation operator in Evolutionary Algorithms. For a special case, we will establish a direct connection between the measure of coupling introduced here and the *modularity matrix elements* in [2,4].

2 Towards a Quantitative Notion for Modularity

In a recent discussion of possible characterizations of modularity by Watson, several illustrative scenarios are presented that highlight different important aspects of the issue [11]. Since we felt it offers many relevant points and fruitful ideas, in the present paper we wish to build upon some of these discussions, offer alternative formal notions of modularity and study some of their conceptual and quantitative properties.

2.1 Probability Notation

We will apply the following notation: for a random variable X we will denote its domain by $\mathcal{X}$ and a concrete sample value of X will be denoted by x. Let now

X, Y be jointly distributed random variables. Let $P(X = x)$ be the probability that X assumes the value $x \in \mathcal{X}$ for which, by abuse of notation, we write instead $p(x)$ wherever this is unambiguous. Similarly, let $P(X = x, Y = y) \equiv p(x, y)$ be the joint and $P(Y = y|X = x) \equiv p(y|x)$ the conditional distribution.

2.2 Watson's Regulatory Network Scenario

One of the scenarios introduced by Watson is a (stochastical) dynamical system which can be seen as a very simple model for a regulatory network [11]. Consider thus a stochastic dynamical system of four random variables S_1, S_2, S_3, S_3, each of them binary valued, i.e. $S_i \in \{0, 1\}$ for $i \in \{1, 2, 3, 4\}$. Write $S = (S_1, S_2, S_3, S_4)$ for the random variable denoting the complete system which therefore can assume 16 states. Write $S(t)$ for the whole system or $S_i(t)$ for a single variable at a time t. Consider the dynamics

$$P\big(S_i(t+1) = 1 \mid S(t) = s(t)\big) = \sum_j w_{ij} s_j(t) \tag{1}$$

$$P\big(S_i(t+1) = 0 \mid S(t) = s(t)\big) = 1 - P\big(S_i(t+1) = 1 \mid S(t) = s(t)\big) \tag{2}$$

where w_{ij} is the weight by which the variable S_j influences S_i. To guarantee well-defined probabilistic expressions, the weights in Eq. (1) are normalized such that $\sum_j w_{ij} = 1$. The discussion by Watson now strives to study the system as composed of two subsystems ("modules"). He achieves this by splitting up the system into two groups of variables (subsystems, "modules"), $M_1 = (S_1, S_2)$ and $M_2 = (S_3, S_4)$ and using different coupling inside and between the subsystems. Watson considers a system with $w_{ij} = 4/10$ if i and j belong to the same subsystem (including $i = j$) and $w_{ij} = 1/10$ if they belong to different subsystems. In this example, increasing coupling reinforces the probability that different S_i will assume the same state. Note that, although designed as two separate coupled subsystems, the question arises here whether M_1 and M_2 can be considered separate subsystems from a *dynamical* point of view.

In principle, $C = 4$ states are possible for each of the subsystems M_1 and M_2. Consider now only systems that converge into a fixed point attractor; there might be several such attractors. Watson distinguishes three kinds of properties for such a system: *non-decomposability, separability* and *decomposability but no separability*. Watson concentrates on the subsystem M_1 and asks about the "most stable states", i.e. the states of M_1 found if the entire system has converged to one of its attractors. Paraphrased from [11], non-decomposability would mean that "for every configuration of M_1 there is some configuration of M_2 (the remainder of the system) that would make that configuration of M_1 the most stable".

In this case, there is a one-to-one relationship between the two subsystems after convergence. Watson characterizes this case by the number C' of possible states of M_1 after the system has converged. In this case $C' = C = 4$, i.e. every possible state of M_1 can be an attractor, depending on the rest. A possible set of attractors fulfilling this property could be $a_1 = (00\,00)$; $a_2 = (01\,01)$; $a_3 = (10\,10)$; $a_4 = (11\,11)$ where the first two bits correspond to M_1 and the last two bits to M_2. In the other extreme, *separability*, "the module is fully independent

in the property of interest" [11]. Watson characterizes this case of separablity by the fact that M_1 can only can have one single state ($C' = 1$). Indeed, in this case (which means that all attractors of the total system are identical w.r.t. the bits of M_1), the attractor state of M_1 does not depend at all on M_2. An example attractor set for this case would consist of $a_1 = (01\,00)$ and $a_2 = (01\,11)$, with M_1 attaining a single value 01. Finally, the intermediate case of *decomposability but non-separability* is now characterized by Watson via $1 < C' < C$. An attractor set showing this property would consist of $a_1 = (11\,11)$ and $a_2 = (00\,00)$ where one has $C' = 2$. As Watson points out, one can see "that there is something we know about the property of interest, the most stable configurations (00 or 11), that is independent of inter-module interactions." [11]

2.3 Open Problems

Although Watson's method of counting the configurations of interest, i.e. the attractors, has some attractive features, two problems remain: 1. The measure is not continous; therefore continuous changes in coupling cannot be measured appropriately. 2. As we will show in the following, the number C' does not always correspond to an intuitive classification of *separablity* or *(non-) decomposability*. Problem 2 can easily be seen by considering the following (not previously discussed) attractor structures:

Case 1: assume the two subsystems were not coupled at all and the corresponding attractors are $a_1 = (00\,00)$; $a_2 = (00\,11)$; $a_3 = (11\,00)$; $a_4 = (11\,11)$ and assumed with equal probability. As the attractors for the individual subsystems are independent of each other, this should be ideally considered as *separable*; in Watson's framework, however, we have $C' = 2$, hence $1 < C' < C$, and thus the system is classified as *decomposable but not separable*. **Case 2:** Assume that each module would converge into all of its 4 configurations. This would be e.g. the case if each binary variable $S_i, i \in \{1, 2, 3, 4\}$ was only fed back to itself in a positive way: Then all 16 combinations could be attractors for the total system. For both modules we would observe $C' = 4$ classifying the system as *non-decomposable*. Intuitively, however, we would classify the system as separable, again as there is no coupling at all between the individual variables and thus between the subsystems.

In the following, we will suggest how to amend these misclassifications as well as how to provide a continuous quantification of Watson's modularity classes of *separablity*, *non-decomposability* and *decomposability but not separablity* by using concepts from information theory.

3 Quantification of Dynamical Modularity

3.1 Information-Theory: Notation

We need some further notions. Define the *entropy* of a random variable X by $H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$ It denotes the expected uncertainty about

a single outcome of X [12]. If the logarithm is chosen w.r.t. base 2, the entropy is quantified by bits; in the following, whereever this value vanishes, we will interchangeably write 0 or 0 bit. Another important quantity is the conditional entropy between two random variables X, Y which is given by $H(Y|X) = \sum_{x\in\mathcal{X}} p(x)H(Y|X = x) = -\sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y|x)\log p(y|x)$ and quantifies how much expected uncertainty in Y remains if X is known. The difference of two related entropies often has the character of entropy or uncertainty loss, or information gain. Specifically, the difference between $H(Y)$ and $H(Y|X)$ is known as the *mutual information* $I(X;Y) = H(Y) - H(Y|X)$ and quantifies how much the knowledge of X adds about the knowledge of Y. The mutual information is symmetric in X and Y and the following relation holds: $I(X;Y) = H(X) + H(Y) - H(X,Y)$. We will use this relation later on.

3.2 An Intuitive Quantification

We will show that mutual information can serve as a modularity measure which is both, intuitive and quantitative. Consider a system with fixed dynamics following Eq. (1) and consider the random variables M_1 and M_2, denoting the first or second subsystem, and taking on values $\{00, 01, 10, 11\}$. Now we measure the coupling by considering the mutual information $I(M_1; M_2)$ which replaces the counting variable C' in Watson's original model. We can now generalize Watson's modularity classes in the following definition.

Definition 1 (Generalized Modularity Classes). *Consider a stochastic dynamical system and assume its attractor is split into an subsystem attractor M_1 and an attractor for the rest M_2. Let $I(M_1; M_2)$ be the mutual information between the two attractor random variables. Then call M_1* separable*, if $I(M_1; M_2) = 0$;* non-decomposable*, if $I(M_1; M_2) = H(M_1)$;* decomposable but not separable*, if $0 < I(M_1; M_2) < H(M_1)$*

Table 1 summarizes the cases we have considered so far. One observes that, while the classification by counting (C') sometimes deviates from what one would intuitively expect, modularity classification based on mutual information meets intuition in all of the cases. It should be mentioned that $I(M_1; M_2)$ does not measure modularity, but its opposite, coupling. A perfectly modular system would be characterized by separability, i.e. $I(M_1; M_2) = 0$.

4 Experiments

It is instructive to see the operation of the measure from Def. 1 in the concrete scenario. For this, consider the dynamical system Eq. (1). For didactical reasons, we use coupling strengths different from [11]: we set $w_{ij} = 1/N$ if i and j belong to the same subsystem and $w_{ij} = c/N$ if they belong to different subsystems (with $N = 2(1 + c)$ a normalization term); $c \in [0, 1]$ is a dynamical coupling strength between the subsystems — if $c = 0$, the subsystems are entirely uncoupled, if $c = 1$, there is no dynamical separation between the subsystems.

Table 1. Summary of the different cases considered with the values for C' and $I(M_1; M_2)$ and the corresponding classifications. In contrast to the classification based on C' (C'-Class), the mutual information $I(M_1; M_2)$ (I-Class) classifies all of the cases according intuition.

Scenario	Intuitive Classif.	C'	C'-Class.	Entropies	$I(M_1; M_2)$	I-Class.
Selecting attractor state M_2, one can obtain any attractor state M_1	non-decomposable (n.-d.)	4	n.-d.	$H(M_1) = 2$ $H(M_2) = 2$ $H(M_1, M_2) = 2$	2	n.-d.
M_1 can only converge to one single state	separable (s.)	1	s.	$H(M_1) = 0$ $H(M_1, M_2) = H(M_2)$	0	s.
M_1 and M_2 can independently converge to one of two attractors **(Case 1)**	separable (s.)	2	d.b.n.s.	$H(M_1) = 1$ $H(M_2) = 1$ $H(M_1, M_2) = 2$	0	s.
M_1 and M_2 converge to one of two attractors, but not independently	decomposable but not separable (d.b.n.s)	2	d.b.n.s.	$H(M_1) = 1$ $H(M_2) = 1$ $0 < H(M_1, M_2) < 2$	> 0 but < 2	d.b.n.s.
Each bit behaves independenlty **(Case 2)**	separable (s.)	4	n.-d.	$H(M_1) = 2$ $H(M_2) = 2$ $H(M_1, M_2) = 4$	0	s.

We run 100,000 independent simulations of the system for different values of the coupling c obtaining empirical distributions for the attractors of the system (as heuristics derived from experiments, we considered a state to be an attractor and ended the run if the system stays in this state for 10 time steps). Several mutual information quantities obtained for these states are shown in Fig. 1. The results show that for small c also the dynamical coupling $I(M_1; M_2)$ is small, as expected, and it grows continuously with growing c (thus fulfilling the continuity requirement from Sec. 2.3). It reaches a value close to 1 bit for $c = 1$ (the fully coupled system) reflecting the fact that in this case the system ends mostly in the equally distributed stochastic "attractors" $S = (0, 0, 0, 0)$ and $S = (1, 1, 1, 1)$[1]. In this case, we do not expect a natural split of the system in two subsystems.

Up to now we have always assumed that we do know the specific decomposition of the system into subsystems by knowing the form of w_{ij} and the particular structure of the network. However, our measure provides us with a

[1] Because of finite-size effects in the dynamics of (1) and (2), other states are also found with some small probability, thus for $c = 1$ the values for $I(M_1; M_2)$ and $I(S_1, S_3; S_2, S_4)$ drop slightly below 1 bit.

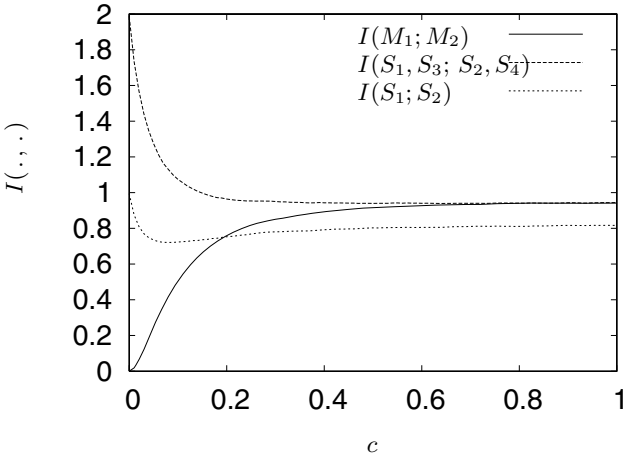


Fig. 1. The plots show the dynamic coupling $I(M_1; M_2)$ between the two subsystems, depending on the coupling constant c (the x-axis); in addition, the mutual information between the crosswise mixed system (S_1, S_3) and (S_2, S_4) as well as the mutual information between the variables (S_1, S_2) inside the system M_1 is shown. For details see text.

way to separate the subsystems without using this knowledge. Plotting the mutual information between alternative subsystems of S, Fig. 1 shows, e.g. the crosswise mutual information $I(S_1, S_3; S_2, S_4)$ arising from an alternative split of S into subsystems. For small c, the value is almost 2 bit, far from 0, indicating that this split does not provide natural modules[2]. On the other hand, once c becomes close to 1, this crosswise information converges towards the value of the coupling information, indicating that the split into S_1, S_2 and S_3, S_4 becomes indistinguishable from S_1, S_3 and S_2, S_4; this is expected, as for $c = 1$ both splits are dynamically equivalent.

Another interesting measure is the intrinsic information in $M_1 = (S_1, S_2)$, i.e. $I(S_1, S_2)$. For $c = 0$, this becomes 1 bit indicating that the variables are fully coupled in the subsystem. Increasing c first slightly reduces the coupling as the subsystem is being perturbed, but then the overall dynamics "stiffens" and again creates a correlation between the variables. However, when $I(M_1; M_2)$ grows, it indicates that modularity is increasingly lost. Once this value becomes larger than $I(S_1, S_2)$, M_1 can be basically seen as having lost its identity as a module, because the coupling between the module and its environment is stronger than the intrinsic coupling. Other combinations can also be compared. This illustrates how the information-theoretic measures provide a whole family of useful characterizations of modularity.

[2] For the extreme case $c = 0$, S_1 is fully aligned with S_2 and S_3 with S_4, thus $I(S_1, S_3; S_2, S_4) = 2$ bit.

5 Applicability to Evolutionary Operators

To round up the argument, we will show that our concepts apply directly to issues of modularity of indidivuals in Evolutionary Algorithms. To this purpose, we consider a measure $m_{jk}(\psi)$ from [2,4] which quantifies the modularity of mating two individuals of given genotypes $j \in \mathcal{S}$ and $k \in \mathcal{S}$, where $\mathcal{S}$ is the search space and ψ is an equivalence relation (more details below). This formalism combines the algebraic formalism from [13] with the dynamical system formalism from [14] and has been used to study under which circumstances self-organization of modularity occurs. In that formalism modularity is always considered w.r.t. a specific equivalence relation[3] ψ defined over the search space $\mathcal{S}$. We will briefly sketch the formalism and show how it fits neatly into the framework developed in the earlier sections. Define then the *modularity matrix element* by

$$m_{jk}(\psi) = \sum_{i \in \mathcal{S}} p(i|j,k)\; r_{i \sim jk}(\psi) \quad . \tag{3}$$

Here $p(i|j,k)$ is the probability that crossover and mutation will generate an offspring individual of type i by mating of individuals of types j and k[4]; $r_{i \sim jk}$ is a binary indicator variable describing whether i is equivalent to one of its parents j or k, i.e. $r_{i \sim jk}(\psi) = 1$ if $i \sim_\psi j$ or $i \sim_\psi k$ and 0 else[5]. Thus, for parents of types j and k, the quantity $m_{jk}(\psi)$ measures the expected degree to which their offspring is equivalent to one of the parents (with respect to ψ). The relation ψ can e.g. be used to model a number of relevant concepts, e.g. traits (like certain phenotypical properties, size, strength, etc.) or schemata [15,16]. It is important to note that the formalism makes no assumption whatsoever about the underlying variation operator(s) or about the representation, e.g. bit-strings, GP-trees etc.

Consider now a simple example of a bitstring-based Genetic Algorithm and two of the bits of the individuals, $b^{(1)}$ and $b^{(2)}$; the genotype of an individual has therefore the following form: $(\ldots b^{(1)} \ldots b^{(2)} \ldots)$. Let an individual of type $j = (\ldots 1 \ldots 1 \ldots)$ and an individual of type $k = (\ldots 0 \ldots 0 \ldots)$ be given. Let furthermore ψ be the equivalence relation which declares two types equivalent iff they are equal in both $b^{(1)}$ and $b^{(2)}$. Then $m_{jk}(\psi)$ becomes the probability that an offspring individual looks completely like j or completely like k with respect to these two bits: $m_{jk}(\psi) = p(\ldots 0 \ldots 0 \ldots | j,k) + p(\ldots 1 \ldots 1 \ldots | j,k)$.

Here, we consider a one-point crossover with $0 \leq p_{\text{crossover}} \leq 0.5$ (for larger crossover probabilities one experiences mirroring effects which we will ignore for the discussion). If we assume symmetry in the variation (i.e. neither $(\ldots 0 \ldots 0 \ldots)$ nor $(\ldots 1 \ldots 1 \ldots)$ is preferred by the variation), the definition of $m_{jk}(\psi)$

[3] An equivalence relation is a relation $\sim$ ("equivalent to") having the properties of reflexivity ($\forall x : x \sim x$), symmetry ($\forall x, y : x \sim y \Leftrightarrow y \sim x$) and transitivity ($\forall x, y, z : x \sim y \wedge y \sim z \Rightarrow x \sim z$).

[4] In [14] and [2,4], $p(i|j,k)$ is called the *transmission function* and written as $T(i \leftarrow j,k)$.

[5] In [2,4], this is denoted as $r_{ijk}(\psi)$.

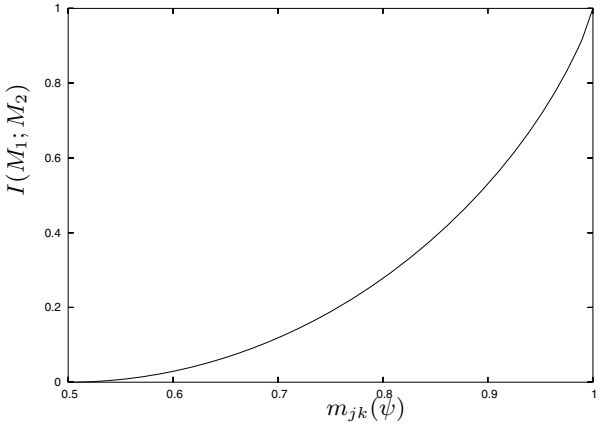


Fig. 2. Relation between $I(M_1; M_2)$ and $m_{jk}(\psi)$ in the example presented

immediately leads to $p(\ldots 0 \ldots 0 \ldots | j, k) = p(\ldots 1 \ldots 1 \ldots | j, k) = \frac{1}{2} m_{jk}(\psi)$ and, with similar assumptions, $p(\ldots 1 \ldots 0 \ldots | j, k) = p(\ldots 0 \ldots 1 \ldots | j, k) = \frac{1}{2}(1 - m_{jk}(\psi))$ holds. Most common crossover operators, as N-point-crossover or uniform crossover fulfill this requirement.

We now create the connection to the approach to measure modularity in the dynamical system. Let B_1 and B_2 be the random variables associated to the probabilities of values for the two bits $b^{(1)}$ and $b^{(2)}$. From the symmetry assumptions, it is clear that

$$P(b^{(i)} = 0) = P(b^{(i)} = 1) = 0.5 \quad i = 1, 2 \tag{4}$$

and therefore $H(B_1) = H(B_2) = 1$. As the above equations show the joint distribution of both bits depends on $m_{jk}(\psi)$, and thus the mutual information. Using above relations for $p(\ldots 0 \ldots 0 \ldots | j, k)$ and $p(\ldots 1 \ldots 0 \ldots | j, k)$, the mutual information can be computed. The functional dependency between $I(B_1; B_2)$ and $m_{jk}(\psi)$ is shown in Fig. 2. It corresponds to the intrinsic information $I(S_1; S_2)$ from Sec. 4 which quantifies how strongly the variables of a specific subsystem are coupled and shows that increasing modularity corresponds to stronger coupling of the intrinsic variables. An information-theoretic measure for inter-module coupling in that scenario can also be formulated, but it is more technically involved and will have to be discussed elsewhere for lack of space.

6 Conclusion and Future Work

We have presented two different directions of thrust towards a formalization of modularity. For this purpose, we have discussed Watson's example of a dynamical system and used information theory to obtain a generalized version of Watson's modularity classes. In particular, we were able to classify properly several cases that are unintuitive under Watson's C' counting approach. In addition, we were

able to reconstruct the modular structure of the regulatory network by looking at the mutual information between different parts of the system (Fig. 1).

In the other line of thrust, we were able to create a connection between the measure $m_{jk}(\psi)$ of modularity by which self-organized modularity in evolving systems has been studied in earlier work and the information-theoretic picture of dynamical systems modularity developed here. This indicates that the present approach can be extended to become a powerful tool to establish when and how modularity arises and, perhaps, to design Alife systems that are able to evolve their own modular decompositions in a targeted manner and thus are able to move more swiftly towards higher rungs in the ladder of complexity.

References

1. W. Zuser, S. Biffl, T. Grechenig, and M. Köhle. *Software Engineering.* Pearson Studium, 2001.
2. P. Dauscher and T. Uthmann. On self-organizing modularization of individuals in evolutionary scenarios. In D. Polani, J. Kim, and T. Martinetz, editors, *Proc. Fifth German Workshop on Artificial Life, March 18-20, 2002, Lübeck, Germany.* Akademische Verlagsgesellschaft Aka GmbH, Berlin, 2002.
3. P. Dauscher. *Selbstorganisierte Modularisierung von Individuen in Evolutionären Algorithmen.* PhD thesis, Johannes Gutenberg-Universität Mainz, 2003.
4. P. Dauscher and T. Uthmann. Self-organized modularity in evolutionary algorithms. *Evolutionary Computation*, 2005. To appear.
5. Stefan Winter. Zerlegung von gekoppelten Dynamischen Systemen (Decomposition of Coupled Dynamical Systems. Diploma thesis, Johannes Gutenberg-Universität Mainz, 1996. (In German).
6. Gustavo Deco and Bernd Schuermann, editors. *Information Dynamics: Foundations and Applications.* Springer, 2001.
7. E. Ziv, M. Middendorf, and C. Wiggins. An information-theoretic approach to network modularity. *Physical Review E*, In press, 2005. arXiv:q-bio.QM/0411033v1.
8. K. A. Eriksen, I. Simonsen, S. Maslov, and K. Sneppen. Modularity and extreme edges of the internet. *Phys. Rev. Lett.*, 90:148701, 2003.
9. P. Magwene. New tools for studying integration and modularity. *Evolution*, 55(9):1734–1745, 2001.
10. Thomas Schreiber. Measuring information transfer. *Phys. Rev. Lett.*, 85:461–464, 2000.
11. R.A. Watson. Modular interdependency in complex dynamical systems. In Bilotta et al., editors, *Workshop Proceedings of the 8th International Conference on the Simulation and Synthesis of Living Systems, UNSW Australia, December 2002*, 2003.
12. Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.
13. Nicholas J. Radcliffe. The Algebra of Genetic Algorithms. *Annals of Maths and Artificial Intelligence*, 10(4), 1994.
14. Lee Altenberg. The evolution of evolvability in genetic programming. In J. K. E. Kinnear, editor, *Advances in Genetic Programming.* MIT Press, 1994.
15. D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, 1989.
16. M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, 1996.

On the Mean Convergence Time of Multi-parent Genetic Algorithms Without Selection

Chuan-Kang Ting

International Graduate School of Dynamic Intelligent Systems,
University of Paderborn, 33100 Paderborn, Germany
ckting@upb.de

Abstract. This paper investigates genetic drift in multi-parent genetic algorithms (MPGAs). An exact model based on Markov chains is proposed to formulate the variation of gene frequency. This model identifies the correlation between the adopted number of parents and the mean convergence time. Moreover, it reveals the pairwise equivalence phenomenon in the number of parents and indicates the acceleration of genetic drift in MPGAs. The good fit between theoretical and experimental results further verifies the capability of this model.

1 Introduction

Multi-parent genetic algorithms (MPGAs) are genetic algorithms using multi-parent crossovers. Traditionally, genetic algorithms (GAs) adopt two parents in crossover to reproduce offspring. This idea is reasonable because, to the best of our knowledge, the form of sexual reproduction on the Earth is absolutely of two parents. Multi-parent crossovers break through this natural limitation by allowing more than two parents in the process of crossover. In a sense, multi-parent genetic algorithms can be viewed as multi-parent generalization of genetic algorithms. In light of MPGAs, several multi-parent crossovers have been proposed and shown their power in a variety of optimization problems [5,7,15]. However, most of these crossovers are validated empirically; only a few theoretical analyses of multi-parent crossovers are conducted.

Genetic drift is a phenomenon that a finite population, even if no selection pressure is applied, will ultimately converge to a uniform population. The rate of genetic drift serves as an important index of how fast population diversity is lost. Schippers [14] studied the genetic drift of two multi-parent crossovers: *uniform scanning crossover* (U-Scan) and *occurrence based scanning crossover* (OB-Scan). His work revealed that U-Scan has no influence on genetic drift whilst OB-Scan induces severe genetic drift, as the number of parents is increased. Nevertheless, the strength of genetic drift in his work is determined by comparing the probabilities of drift in and drift out. The rate of genetic drift in MPGAs is still an open question.

This paper investigates in theory the rate of genetic drift in MPGA using OB-Scan. Specifically, we propose an exact model for the mean convergence

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 403–412, 2005.

time — the principal measure of genetic drift rate [2]. First, we analyze the gene frequency altered by multi-parent crossovers. Based on gene frequency, we model the behavior of MPGAs through Markov chains. This Markov model affords the correlation between parent numbers and mean convergence time. In addition, the theoretical results reveal the pairwise equivalence of parent numbers in OB-Scan. These theoretical results are further verified by a series of experiments.

The rest of this paper is organized as follows. Section 2 describes OB-Scan and Section 3 analyzes its effect on gene frequency. Next, we model MPGAs with Markov chains. Theoretical results and experimental validation are presented in Section 5. Finally, conclusions are drawn in Section 6.

2 Occurrence Based Scanning Crossover (OB-Scan)

Occurrence based scanning crossover is one of the *scanning crossovers* proposed by Eiben et al. [6]. Scanning crossovers are multi-parent generalization of *uniform crossover* — a widely used crossover in GAs. In uniform crossover, the donor for each locus is randomly picked from two selected parents. Extended to more than two parents, scanning crossovers choose the donor at random or using heuristics. According to different strategies, Eiben et al. proposed three variations of scanning crossovers: *uniform scanning crossover* (U-Scan), *occurrence based scanning crossover* (OB-Scan), and *fitness based scanning crossover* (FB-Scan). In this paper, we only discuss OB-Scan.

Rather than at random, OB-Scan determines offspring genes depending on the occurrence of parental genes at that locus. Specifically, it picks the *majority* of parental gene values as the offspring gene for each locus. Note that in this paper OB-Scan is defined to break ties by randomly[1] choosing a binary. Examples of 2-parent OB-Scan (corresponding to uniform crossover) and 4-parent OB-Scan are given in Fig. 1. The formal definitions of the components of GAs and OB-Scan are drawn below.

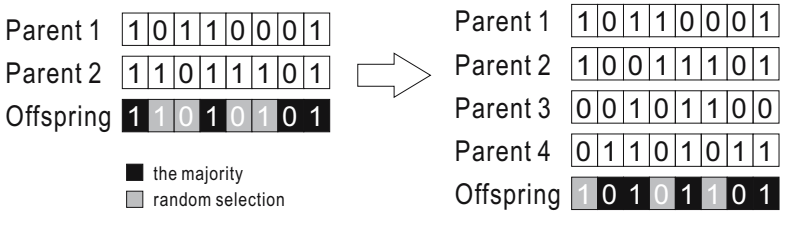


Fig. 1. Examples of 2-parent OB-Scan (left) and 4-parent OB-Scan (right)

[1] In the original version of OB-Scan [6], OB-Scan breaks ties by directly inheriting the genes of the first selected parent. However, random tie break conforms to generalization of uniform crossover.

Definition 1 (Chromosome and Population).

1. *A* chromosome $\boldsymbol{c}$ *is encoded as a bit string, i.e.* $\boldsymbol{c} \stackrel{\text{def}}{=} (c_1, \ldots, c_l) \in \{0,1\}^l$, *where* c_i *denotes a* gene *and* l *is the* chromosome length.
2. *A* population C *is a set of chromosomes:* $C \stackrel{\text{def}}{=} \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m\}$, *where* $\boldsymbol{c}_i \in \{0,1\}^l$ *and* m *is the* population size.

Definition 2 (OB-Scan). *Given* n *parents* $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n \in C$ *selected from population* C, *OB-Scan reproduces the offspring* $\boldsymbol{c}' = \mathcal{X}_{\text{ob}}(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n) = (c'_1, \ldots, c'_l)$ *by*

$$c'_i = \begin{cases} 0 & \text{if } \sum_{j=1}^{n} (\boldsymbol{c}_j)_i < \frac{n}{2} \\ 1 & \text{if } \sum_{j=1}^{n} (\boldsymbol{c}_j)_i > \frac{n}{2} \\ \text{rand}(0,1) & \text{otherwise} \end{cases} \quad \text{for } i = 1, \ldots, l,$$

where $(\boldsymbol{c}_j)_i$ *denotes the* i^{th} *gene of the chromosome* $\boldsymbol{c}_j$, *and* rand $(0,1)$ *is a binary random function.*

3 Variation of Gene Frequency

Gene frequency is widely used as a quantitative measure of genetic variation in population genetics [9]. It also suffices to clue us in on the course of evolution in GAs. In this section we analyze the variation of gene frequency caused by OB-Scan.

Definition 3 (Gene Frequency). *The* gene frequency $p_k(\alpha, t)$ *is defined as the proportion of allele* α *at locus* k *in the population at time* t. *Let* $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_m\}$ *be a population at time* t *and let* $C_k(\alpha) = \{\boldsymbol{c} \in C \mid c_k = \alpha\}$ *be the subset of population where chromosomes possess allele* α *at locus* k. *The gene frequency*

$$p_k(\alpha, t) \stackrel{\text{def}}{=} \frac{|C_k(\alpha)|}{|C|},$$

where $|C|$ *and* $|C_k(\alpha)|$ *represent the cardinality of set* C *and* $C_k(\alpha)$.

As above defined, chromosomes are represented as binary strings. Thus there exists exactly two gene frequencies $p_k(1,t)$ and $p_k(0,t)$ with $p_k(1,t) = 1 - p_k(0,t)$ for every locus k and time t. For simplicity, we refer to the gene frequency $p_k(1,t)$ as $p_k(t)$ and refer to $p_k(0,t)$ as $(1 - p_k(t))$.

Definition 4 (Variation of Gene Frequency in GAs). *Let* $p_k^s(t), p_k^x(t), p_k^m(t)$ *be the gene frequencies after performing selection, crossover, and mutation at generation* t. *The variation of gene frequency in GAs can be expressed as*

$$p_k(t) \xrightarrow{\text{selection}} p_k^s(t) \xrightarrow{\text{crossover}} p_k^x(t) \xrightarrow{\text{mutation}} p_k^m(t) \xrightarrow{\text{survivor}} p_k(t+1), \qquad (1)$$

To investigate genetic drift, *random selection* and *no mutation* is assumed. In this paper we focus on generational GAs. As a result, the gene frequency in

the process of GA is only affected by the sampling of random selection and the process of crossover. These influences will be analyzed in Lemma 1. Incidentally, the symbol $p_k(t)$ is referred to as p_k while the indication of time t is not in effect.

Before conducting the analysis of OB-Scan, we introduce the incomplete beta function for simplifying the expression of equations .

Definition 5 (Incomplete Beta Function). *The* incomplete beta function *is defined as*

$$I_x(a,b) \stackrel{\text{def}}{=} \frac{1}{\text{Beta}(a,b)} \int_0^x t^{a-1}(1-t)^{b-1}dt,$$

where $a, b > 0$ *and* $\text{Beta}(a,b)$ *is the* beta function.

The incomplete beta function holds the following properties:

1. (26.5.24 [1]) For binomial distribution $B(n,p)$,

$$\sum_{i=a}^{n} B(i;n,p) = I_p(a, n-a+1). \tag{2}$$

2. (26.5.16 [1])

$$I_x(a,b) = \frac{1}{a \cdot \text{Beta}(a,b)} x^a (1-x)^b + I_x(a+1,b). \tag{3}$$

Using the above definition and properties, we embark on the analysis of OB-Scan's influence on gene frequency.

Lemma 1. *Suppose we have the gene frequency* p_k *of the population. The gene frequency, denoted by* p_k^{ob}*, of the offspring reproduced by* n*-parent OB-Scan* $\mathcal{X}_{\text{ob}}$ *with* $n \in \mathbb{N}_{>1}$ *is*

$$p_k^{\text{ob}} = I_{p_k}(a,a),$$

where I_p *denotes the incomplete beta function and* $a = \left\lceil \frac{n}{2} \right\rceil$.

Proof. Let X be the number of parents possessing the allele 1 at locus k among n selected parents. Since the process of random selection is independent, it is a Bernoulli process. Performing this selection n times, the number X holds a binomial distribution with probability mass function

$$\Pr(X = x) = B(x;n,p_k) = \binom{n}{x} (p_k)^x (1-p_k)^{n-x}.$$

Let $\mathfrak{D}_1$ denote the event that OB-Scan assigns the allele 1 to the offspring locus k. According to Definition 2, OB-Scan yields

$$\Pr(\mathfrak{D}_1 \mid X = x) = \begin{cases} 1 & \text{if } x > n/2, \\ 0 & \text{if } x < n/2, \\ \frac{1}{2} & \text{if } x = n/2. \end{cases}$$

For OB-Scan with an odd number of parents ($n = 2a - 1$ with $a \in \mathbb{N}_{>1}$), from (2) we have

$$\begin{aligned} p_k^{\mathrm{ob}} = \Pr(\mathfrak{D}_1) &= \sum_{x=0}^{n} \Pr(\mathfrak{D}_1 \mid X = x) \cdot \Pr(X = x) \\ &= \sum_{x=a}^{2a-1} B(x; 2a-1, p_k) = I_{p_k}(a, a). \end{aligned}$$

Similarly, for OB-Scan with an even number of parents ($n = 2a$ with $a \in \mathbb{N}$),

$$\begin{aligned} p_k^{\mathrm{ob}} &= \sum_{x=a+1}^{2a} B(x; 2a, p_k) + \frac{1}{2} B(a; 2a, p_k) \\ &= I_{p_k}(a, a) - \frac{1}{a\mathrm{Beta}(a, a)} (p_k)^a (1 - p_k)^a + \frac{1}{2} \binom{2a}{a} (p_k)^a (1 - p_k)^a \\ &= I_{p_k}(a, a) + \left[-\frac{\Gamma(2a)}{a\Gamma(a)\Gamma(a)} + \frac{1}{2} \frac{(2a)!}{a!a!} \right] (p_k)^a (1 - p_k)^a \\ &= I_{p_k}(a, a). \end{aligned}$$

□

Corollary 1 (Pairwise Equivalence). *Let $p_k^{\mathrm{ob}(n)}$ be the gene frequency p_k^{ob} corresponding to n-parent OB-Scan. For $n \in 2\mathbb{N}$ and $n \geq 4$,*

$$p_k^{\mathrm{ob}(n)} = p_k^{\mathrm{ob}(n-1)}.$$

Proof. Trivial (since $\left\lceil \frac{n}{2} \right\rceil = \left\lceil \frac{n-1}{2} \right\rceil$ in Lemma 1). □

4 Modeling with Markov Chains

Markov chains have been used to model the exact behavior of GAs [2,8,10] and to analyze the convergence of GAs [4,11,13]. In this paper, we use Markov chains to model the evolution of gene frequency. From this Markov model, the mean convergence time can be derived.

In light of gene frequency, a GA can be viewed as a stochastic process manipulating the number of allele 1 (or 0) in the population: Let random variables $G_k(t) \in \{0, 1, \ldots, m\}$ be the number of allele 1 at locus k at generation t. The process of GAs on gene frequency can be represented as $\{G_k(t) : t \in \mathbb{Z}_*\}$. Since for every $i_0, i_1, \cdots, i_{t+1} \in \{0, 1, \ldots, m\}$ the process $\{G_k(t)\}$ satisfies

$$\begin{aligned} \Pr\{G_k(t+1) = i_{t+1} \mid G_k(t) = i_t, G_k(t-1) = i_{t-1}, \ldots, G_k(0) = i_0\} \\ = \Pr\{G_k(t+1) = i_{t+1} \mid G_k(t) = i_t\}, \end{aligned}$$

the process $\{G_k(t)\}$ is a Markov chain. The formal definition of the Markov chain for gene frequency is given as follows.

Definition 6 (Markov Chain for Gene Frequency). *In the Markov chain* $\{G_k(t)\}$ *for gene frequency at locus* $k \in \{1, \ldots, l\}$ *in GAs,*

1. *the* state *is defined as the number of allele* 1 *at locus* k *in the population and the state space is thus* $\{0, 1, \ldots, m\}$. *A state* i *in* $\{G_k(t)\}$ *implies the gene frequency at locus* k *is*
$$p_k = \frac{i}{m}.$$
2. *The* transition matrix *of* $\{G_k(t)\}$ *is defined as* $\mathbf{P} \stackrel{\text{def}}{=} (\rho_{ij})$, *where* ρ_{ij} *is the* transition probability *of state* i *to state* j*:*
$$\rho_{ij} \stackrel{\text{def}}{=} \Pr\{G_k(t+1) = j \mid G_k(t) = i\}.$$

The previous section has shown how OB-Scan changes the gene frequency. From those formulae, we derive the transition probabilities of the Markov chain $\{G_k(t)\}$ for the evolution of gene frequency in MPGAs.

Theorem 1. *For a GA using random selection,* n*-parent OB-Scan, and no mutation, the transition probability* ρ_{ij} *of the Markov chain* $\{G_k(t)\}$ *corresponding to this GA is*
$$\rho_{ij} = B(j; m, p'_k)$$
with
$$p'_k = I_{\frac{i}{m}}\left(\left\lceil \frac{n}{2} \right\rceil, \left\lceil \frac{n}{2} \right\rceil\right).$$

Proof. The state i of transition probability ρ_{ij} suggests the gene frequency $p_k = \frac{i}{m}$. Given this frequency p_k, from Lemma 1 we can obtain the gene frequency p'_k of the offspring reproduced by a GA using random selection, n-parent OB-Scan, and no mutation:
$$p'_k = p_k^{\text{ob}} = I_{\frac{i}{m}}\left(\left\lceil \frac{n}{2} \right\rceil, \left\lceil \frac{n}{2} \right\rceil\right).$$
In generational GAs, population is completely replaced with the subpopulation, consisting of m offspring reproduced by m times of *selection-crossover-mutation* process. Since this process is independent, the number of allele 1 holds a binomial distribution $B(m, p'_k)$. Therefore, the transition probability
$$\begin{aligned}\rho_{ij} &= \Pr\{G_k(t+1) = j \mid G_k(t) = i\} \\ &= B(j; m, p'_k).\end{aligned}$$
□

For the Markov chain $\{G_k(t)\}$, of particular interest to us is, if at all, the *convergence* of $\{G_k(t)\}$ — at that time the population turns out to be either all-zeros or all-ones at each locus. For this, first we introduce a special kind of Markov chains, called *absorbing* Markov chains [3], which have this convergence property, i.e. absorption. Next, we will prove the Markov chain corresponding to the aforementioned GA belongs to such kind of Markov chains; that is to say, the corresponding GA will converge. From the properties of absorbing Markov chains we can further derive the mean convergence time.

Definition 7 (Absorbing States).

1. *The* closed set S^c *is a set of states whose transition probabilities* $\rho_{ij} = 0$ *for all* $i \in S^c, j \notin S^c$.
2. *A state* i *is said to be* absorbing *if and only if* $\exists S^c : S^c = \{i\} \Longleftrightarrow \rho_{ii} = 1$.
3. *A Markov chain with absorbing states is called an absorbing Markov chain.*

Proposition 1. *For the GA given in Theorem 1, its corresponding Markov chain* $\{G_k(t)\}$ *is an absorbing Markov chain with exactly two absorbing states:* 0 *and* m.

Proof. According to the definition of absorption and Theorem 1, we know

$$\{G_k(t)\} \text{ is absorbing } \Longleftrightarrow \exists i : \rho_{ii} = 1 \Longleftrightarrow \exists i : B(i; m, p'_k) = 1. \tag{4}$$

The solutions of $B(i; m, p'_k) = 1$ subject to $p'_k = I_{\frac{i}{m}}\left(\left\lceil \frac{n}{2} \right\rceil, \left\lceil \frac{n}{2} \right\rceil\right)$ and $n \in \mathbb{N}_{>1}$ are (i) $i = 0$ with $p'_k = 0$ and (ii) $i = m$ with $p'_k = 1$. This leads to, for the Markov chain $\{G_k(t)\}$

$$\rho_{00} = \rho_{mm} = 1 \implies \{G_k(t)\} \text{ is absorbing .}$$

Thus we complete the proof that the Markov chain $\{G_k(t)\}$ is absorbing with exactly two absorbing states 0 and m. □

The above proposition indicates the chain $\{G_k(t)\}$ will get absorbed into state 0 or m. It means that the process of the predefined GA will 'drift' into all-zeros or all-ones for each locus, that is, reaching convergence. In addition to the existence of convergence, we are interested in the mean time to reach it. To compute the mean time for a chain to get absorbed, we introduce the fundamental matrix [3] and its related property below.

Definition 8 (Fundamental Matrix). *For a Markov chain with* b *absorbing states, the transition matrix can be rewritten as*

$$\mathbf{P} = \begin{pmatrix} \mathbf{I}_b & \mathbf{0} \\ \mathbf{R} & \mathbf{Q} \end{pmatrix}, \tag{5}$$

where $\mathbf{I}_b$ *is a* $b \times b$ *identity matrix. The* fundamental matrix *for this absorbing Markov chain is defined as*

$$\mathbf{F} \stackrel{\text{def}}{=} (\mathbf{I} - \mathbf{Q})^{-1}.$$

Theorem 2 ([12]). *Let* $\mathbf{F}$ *be the fundamental matrix of an absorbing Markov chain. The fundamental matrix* $\mathbf{F}$ *stands for the mean time* τ_{ij} *that the process spends at transient state* j *starting from transient state* i.

Theorem 3 (The Mean Convergence Time of MPGA). *Suppose we have the GA given in Theorem 1. Let* $\mathbf{F} = (\tau_{ij})$ *be the fundamental matrix of the Markov chain* $\{G_k(t)\}$ *corresponding to this GA. For some locus* $k \in \{1, \ldots, l\}$, *given the initial state distribution* $\boldsymbol{\pi}(0) = (\pi_0(0), \ldots, \pi_m(0))$, *the mean convergence time*

$$\tau = \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \pi_i(0)\tau_{ij}.$$

Proof. Let A be the set of absorbing states in $\{G_k(t)\}$. Proposition 1 gives $A \equiv \{0, m\}$. According to Theorem 2, the fundamental matrix of $\{G_k(t)\}$ represents the mean time τ_{ij} for $i, j \in \overline{A}$. Hence, the mean time τ_i that the process spends among transient states starting from transient state i can be derived by

$$\tau_i = \sum_{j \in \overline{A}} \tau_{ij}.$$

Given the initial state distribution $\boldsymbol{\pi}(0)$, we have the mean convergence time

$$\tau = \sum_{i=1}^{m-1} \tau_i \Pr(i \mid t = 0) = \sum_{i=1}^{m-1} \tau_i \pi_i(0) = \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \pi_i(0) \tau_{ij}. \qquad \square$$

5 Theoretical Results and Experimental Validation

This section demonstrates theoretical results obtained from the above theorems. Moreover, we conduct experiments on single locus ($l = 1$) to verify these theoretical results. The setting of MPGA used in our experiments is generational GA, bit-string representation, random selection, and no mutation. Each experiment setting includes 1000 independent runs.

Figure 2 compares the mean convergence time obtained from Theorem 3 and from experiments. First, this figure shows that the theoretical and the experimental results fit very well. In addition, it shows that for $n \in 2\mathbb{N}$ a MPGA using n-parent OB-Scan performs correspondingly to that using $(n-1)$-parent OB-Scan, which confirms the *pairwise equivalence* claimed in Corollary 1. Second, this figure indicates the fact that, compared with two parents, using more than two parents in OB-Scan causes a drastic decrease in mean convergence

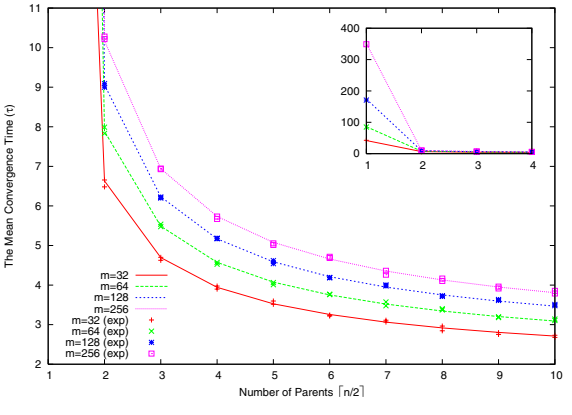


Fig. 2. Comparison of the theoretical (lines) and the experimental (symbols) mean convergence time τ for n-parent OB-Scan and population size m

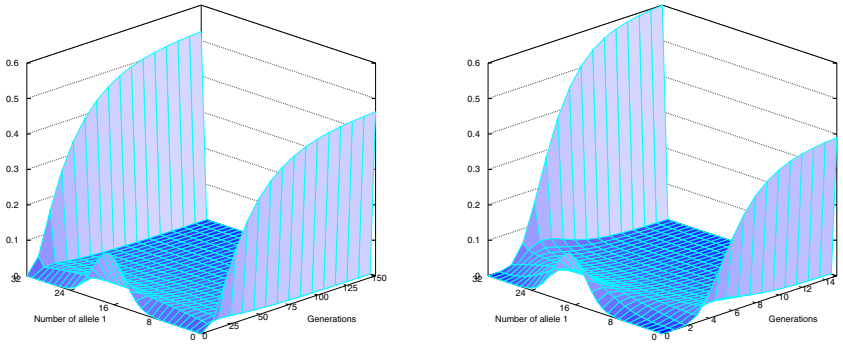


Fig. 3. The progress of genetic drift of 2-parent OB-Scan (left) and 3-parent OB-Scan (right) for population size $m = 32$ with initialization bias $\beta = \frac{1}{32}$

time. Nonetheless, further raising parents yields a relatively small difference. The mean convergence time ($m = 256$) for $n = 2$, for example, needs 351.55 generations while it takes only 10.17 generations for $n = 3$ (or 4) and 6.94 generations for $n = 5$ (or 6). This speedup in convergence reflects that OB-Scan with more than two parents accelerates genetic drift.

Next, we examine the progress of genetic drift in case of a initialization bias. We denote by β the bias of initial gene frequency to the allele 1. Figure 3 compares the progress of genetic drift of uniform crossover (i.e. 2-parent OB-Scan) and 3-parent OB-Scan for population size $m = 32$ under initialization bias $\beta = \frac{1}{32}$. As aforementioned, the genetic drift of 3-parent OB-Scan is much faster than that of uniform crossover. Interestingly, the distribution of convergence probability of uniform crossover differs from that of 3-parent OB-Scan either. Asoh and Mühlenbein [2] have shown the convergence probability of uniform crossover equals the initialization probability, which is reflected in Fig. 3. Yet, adopting more parents does not follow this rule. The 3-parent OB-Scan gives a probability (≈ 0.6) higher than the initialization probability ($\frac{17}{32} \approx 0.53$). This outcome suggests that using more parents in OB-Scan will intensify the preference of the initialization.

6 Conclusions

This paper presents an exact model for exploration of genetic drift in MPGAs. First we analyze the gene frequency altered by OB-Scan. Based on gene frequency, we model the behavior of MPGAs through Markov chains. The mean convergence time is further derived from this model.

The theoretical results demonstrate that raising parents in OB-Scan shortens the mean convergence time; that is, it accelerates genetic drift. This outcome not only reconfirms Schippers' claims about the genetic drift of scanning crossovers, but also gives the expected time of convergence. In addition, our analysis reveals the pairwise equivalence in OB-Scan: n-parent OB-Scan performs analogously

with $(n-1)$-parent OB-Scan for $n \in 2\mathbb{N}$. Moreover, the progress of genetic drift under initialization bias suggests raising parents in OB-Scan will intensify the preference of initialization for allele 0 or 1. The good fit between theoretical and experimental results validates our theoretical arguments and the capability of the proposed model.

References

1. M. Abramowitz and I.A. Stegun, editors. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, 1972. ninth Dover printing.
2. H. Asoh and H. Mühlenbein. On the mean convergence time of evolutionary algorithms without selection and mutation. In *Parallel Problem Solving from Nature – PPSN III*, volume 866 of *LNCS*, pages 88–97, Berlin, 1994. Springer.
3. P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer Verlag, 1999.
4. T.E. Davis and J.C. Principe. A markov chain framework for the simple genetic algorithm. *Evolutionary Computation*, 1(3):269–288, 1993.
5. A.E. Eiben. Multiparent recombination in evolutionary computing. In *Advances in Evolutionary Computing*, pages 175–192. Springer, 2002.
6. A.E. Eiben, P.-E. Rau'e, and Zs. Ruttkay. Genetic algorithms with multi-parent recombination. In *Parallel Problem Solving from Nature – PPSN III*, volume 866 of *LNCS*, pages 78–87. Springer, 1994.
7. A.E. Eiben and C.H.M. van Kemenade. Diagonal crossover in genetic algorithms for numerical optimization. *Journal of Control and Cybernetics*, 26(3):447–465, 1997.
8. D.E. Goldberg and P. Segrest. Finite markov chain analysis of genetic algorithms. In *Proceedings of the 2nd International Conference on Genetic Algorithms and their Applications*, pages 1–8. Lawrence Erlbaum Associates, 1987.
9. D.L. Hartl and A.G. Clark. *Principles of Poppulation Genetics*. Sinauer Associates, 1 edition, 1989.
10. J. Horn. Finite markov chain analysis of genetic algorithms with niching. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 110–117. Morgan Kaufmann Publishers, 1993.
11. A.E. Nix and M.D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5:78–88, 1992.
12. A. Papoulis and S.U. Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York, NY, USA, 4 edition, 2002. ISBN 0-07-048477-5.
13. G. Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1):96–101, 1994.
14. C.A. Schippers. Multi-parent scanning crossover and genetic drift. In *Theoretical Aspects of Evolutionary Computing*, pages 307–330. Springer, 2001.
15. S. Tsutsui and A. Ghosh. A study on the effect of multi-parent recombination in real coded genetic algorithms. In *Proceedings of International Conference on Evolutionary Computation*, pages 828–833, 1998.

The Quantitative Law of Effect is a Robust Emergent Property of an Evolutionary Algorithm for Reinforcement Learning

J.J McDowell[1] and Zahra Ansari

Department of Psychology, Emory University, Atlanta, Georgia, U.S.A., 30322
jack.mcdowell@emory.edu, zansari@learnlink.emory.edu

Abstract. An evolutionary reinforcement-learning algorithm, the operation of which was not associated with an optimality condition, was instantiated in an artificial organism. The algorithm caused the organism's behavior to evolve in response to selection pressure applied by reinforcement from the environment. The resulting behavior was consistent with the well-established quantitative law of effect, which asserts that the time rate of a behavior is a hyperbolic function of the time rate of reinforcement obtained for the behavior. The high-order, steady-state, hyperbolic relationship between behavior and reinforcement exhibited by the artificial organism did not depend on specific qualitative or quantitative features of the evolutionary algorithm, and it described the organism's behavior significantly better than other, similar, function forms. This evolutionary algorithm is a good candidate for a dynamics of live behavior, and it might be a useful building block for more complex artificial organisms.

1 Background: Matching Theory and Reinforcement Learning

During the past three decades, the mathematical description of behavior-environment relationships has become an important part of the experimental analysis of behavior. Perhaps the most widely studied and successful mathematical work in behavior analysis is the family of equations known as matching theory [1]. In dozens of experiments with many species, including humans, matching theory has been shown to accurately describe the relationship between properties of behavior and properties of a variety of psychologically significant environments. The most fundamental equation of matching theory is its hyperbolic rate equation, which is often referred to as the quantitative law of effect.

As is well known, E. L. Thorndike (c. 1911) discovered the law of effect, or principle of reinforcement, in his famous puzzle-box experiments. B. F. Skinner (c. 1938) later gave the law of effect a stochastic cast by stating that positive reinforcement increased the probability of a behavior's future occurrence. In 1961, Skinner's student, R. J. Herrnstein, published an influential paper [3] in which he reported that pigeons' rates of choosing various alternatives (i.e., keys to peck) in a multi-alternative

[1] Presentation of this paper was aided by a faculty travel grant to the first author from the Institute of Comparative and International Studies, Emory University.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 413 – 422, 2005.

environment was governed with a remarkable degree of accuracy and precision by a simple algebraic equation that related the rate of key pecking on the various alternatives to the rate of reinforcement obtained for pecking on those alternatives. This equation came to be known as the matching law. From this equation, Herrnstein [4] derived the hyperbolic rate equation in 1970, and since then a great deal of experimental and mathematical research on matching theory has expanded its scope to many species, behaviors and reinforcers, and to a variety of experimental and naturalistic environments [1].

The hyperbolic rate equation, or quantitative law of effect, states how the absolute rate of a behavior, R, in a given environment is governed by the absolute rate of reinforcement, r, obtained for that behavior,

$$R = \frac{kr}{r + r_e}, \tag{1}$$

where k and r_e are parameters of the hyperbola. The parameter, k, is the y-asymptote of the hyperbola, and r_e determines its curvature, that is, how rapidly the function approaches its asymptote. As interpreted by matching theory, k is related to properties of behavior such as the amount of effort the behavior requires, and r_e is related to additional sources of reinforcement that may be available in the environment. In behavior analysis, Equation 1 is now recognized as a fundamental statement of the way reinforcement governs behavior.

An important feature of Equation 1 is that it describes behavior in the steady state, when it is in equilibrium with conditions in the environment. Each point on the hyperbola represents, for a particular behavior and a particular reinforcer, the average equilibrium response rate that is supported by an average reinforcement rate. In most experimental situations, R >> r, in other words, relatively few instances of the behavior are reinforced. The problem of how behavior gets to the steady state has been pursued vigorously, but as yet has not yielded a generally accepted mathematical dynamics. As might be supposed, one of the most popular approaches to this problem is based on optimality theory [9]. Another approach is based on linear filtering [7], and very recent work has made use of computational modeling based on an evolutionary algorithm [6]. The computational approach to behavioral dynamics, which dovetails with work on reinforcement learning in artificial life and related disciplines, is the subject of this article.

Reinforcement learning algorithms in machine learning and artificial intelligence fall into two broad categories. Algorithms in one category deal with the expected utility or value of different courses of action [5, 12]. Temporal-difference learning is an example of this type of algorithm. Utility-based algorithms have been applied to many problems, including some that are relevant to the behavior of live organisms, such as chaining [13], conditioned reinforcement [13], and multi-alternative responding that is consistent with Herrnstein's original matching equation [2]. The second category of reinforcement learning algorithms is concerned with finding the best action or policy in a particular set of circumstances [8]. These algorithms usually entail evolutionary principles. Action-based evolutionary algorithms have also been widely applied, including to problems that are relevant to the behavior of live organisms, such as foraging in multi-alternative environments, which can also be described by the original

matching equation [10-11]. Virtually all of the existing utility-based and action-based reinforcement learning algorithms are designed to solve an optimality problem, that is, they work either by attempting to maximize the expected utility of a sequence of actions, or by attempting to maximize in some way the overall outcomes of an agent's actions.

The reinforcement learning algorithm that will be discussed in this article falls into the second category, although it is not a typical example of this category. It is an evolutionary algorithm that is not, however, designed to solve an optimality problem. Instead, it is simply used as the dynamic mechanism of an artificial organism's behavior. The organism's behavior evolves through a process of selection, reproduction and mutation over many generations, or time steps, where selection pressure is applied by the environment in the form of reinforcing stimuli. The behavior reaches steady states in response to constant time-averaged reinforcement rate inputs, and these steady states can be compared to the requirements of Equation 1. The questions of interest in this research are whether the behavior of an artificial organism that operates according to evolutionary principles conforms to Equation 1, and if so, whether this conformance depends on specific implementations of the evolutionary principles.

2 The Artificial Organism and Evolutionary Algorithm

In this section, the structure and operation of the artificial organism will be described, along with the components of the evolutionary algorithm that constitutes its dynamics.

2.1 The Artificial Organism

The artificial organism consists of 100 10-bit strings that represent integers ranging from 0 through 1023. This collection of bit strings constitutes the organism's repertoire of behaviors or actions. The behaviors can be sorted into classes, called operants, based on how they act upon the environment. A rat's or human's lever press in an experimental chamber, for example, is an operant defined by a switch closure. Individual members of this class include a lever press with the right limb, a lever press with the left limb, a high-force press that exceeds the force required for switch closure, and so on. Partitioning the 100 bit strings into operant classes sets the baseline structure and operation of the artificial organism. For our purposes we will define just two classes, one consisting of the 41 integers from 0 through 40, and one consisting of the remaining 983 integers. The first behavioral class will be designated the target operant, analogous to a lever press. The second behavioral class represents doing something else.

The artificial organism is initialized with 100 10-bit strings selected at random from the 1024 possible strings. The organism's behavior at each time step is determined by the relative frequencies with which the integer values of these strings fall into the different operant classes. The relative frequencies constitute the probabilities that the organism will emit a behavior from each class, and these probabilities are used to determine which operant the organism emits at each moment.

2.2 Fitness

When an operant is reinforced, it is identified as fit with respect to conditions in the environment. Two definitions of fitness will be considered. For *midpoint fitness*, the integer midpoint of the reinforced class of behavior is taken as the fitness criterion, that is, it represents the fittest individual behavior. For *specific individual fitness*, the fitness criterion is the integer value of a specific individual behavior selected from the reinforced class, based on the relative frequencies of the individual members of that class. In both cases, the fitness of each of the 100 bit strings that constitute the organism's behavioral repertoire is defined as the absolute value of the difference between that bit string's integer value and the fitness criterion. Note that this method of defining fitness means that lower fitness values are associated with fitter individual behaviors.

2.3 Parents

Following a reinforcement, parents are chosen for mating on the basis of their fitness by selecting fitness values from a uniform fitness density function,

$$p(x) = \frac{1}{2\mu} \qquad \text{for} \quad 0 \le x \le 2\mu\,, \tag{2}$$

a linear fitness density function,

$$p(x) = -\frac{2}{9\mu^2}x + \frac{2}{3\mu} \qquad \text{for} \quad 0 \le x \le 3\mu\,, \tag{3}$$

or an exponential fitness density function,

$$p(x) = \frac{1}{\mu} e^{-\frac{1}{\mu}x}\,. \tag{4}$$

For all functions, $p(x)$ is the probability density associated with a fitness value, x, and μ is the mean of the density function. These fitness density functions are completely determined by their means. They associate higher probability densities with lower fitness values, and hence with fitter individual behaviors. A general method for constructing functions of this type is given in [6].

Following a reinforcement, a father behavior is chosen from the repertoire by drawing a fitness value at random from one of the fitness density functions, and then searching the organism's repertoire for a behavior with that fitness. If none is found, then another fitness value is drawn at random from the fitness density function, and so on, until a father behavior is found. A distinct mother behavior is obtained in the same way.

In the event that reinforcement does not occur at a given time step, parents are selected at random from the organism's repertoire. In either case, 100 sets of parents are chosen, each of which produces one child behavior. The resulting set of 100 child be-

haviors then replaces the artificial organism's behavioral repertoire, and a behavior from this repertoire is chosen for emission using the method described earlier.

2.4 Reproduction

Two types of reproduction will be considered. In *bitwise* reproduction, each bit in a child's bit string is set equal to the corresponding bit either from the father's bit string or from the mother's bit string, each with a probability of 0.5. In *crossover* reproduction, the parents' bit strings are sliced at a random location and then combined by crossing over. One of the resulting bit strings is chosen at random as the child.

2.5 Mutation

After a new generation of behaviors has been produced, a fixed percentage of the behaviors undergoes mutation, that is, the behaviors' integer values are changed. The individual behaviors that undergo mutation are chosen at random from the organism's repertoire. Three methods of mutation will be considered. In *Gaussian* mutation, the integer value of the chosen behavior is taken as the mean of a Gaussian distribution of integers with a specific standard deviation. A value chosen at random from this distribution is then taken as the mutant. Should the mutant fall outside the range of acceptable values (0-1023), it is wrapped to the other end of the range. In *bit-flip* mutation, one bit from the chosen behavior's bit string, selected at random, is changed. In *random individual* mutation, the integer value of the chosen behavior is replaced with a value selected at random from the range, 0-1023.

3 Experimental Studies of the Artificial Organism's Behavior

Extensive parametric studies of the artificial organism's behavior have been conducted [6], and will be summarized here. The purpose of these studies was to determine whether the behavior of the artificial organism conformed to Equation 1, and if so, whether this conformance depended on specific implementations of the rules of the evolutionary algorithm. In all experiments, reinforcement was set up, or made available, at random times following the delivery of the previous reinforcement. Once reinforcement was made available, it was delivered as soon as the organism emitted the target operant. Environments that work in this way are said to arrange random interval (RI) schedules of reinforcement. An RI schedule is characterized by the mean of its intervals. Evidently, an RI schedule with a small mean arranges frequent reinforcement for the target operant, whereas an RI schedule with a large mean arranges infrequent reinforcement.

In the three series of experiments to be described in this section, the mean of the RI schedules ranged from 1 to 200 time ticks. A single experiment consisted of arranging a series of approximately 10 RI schedules, each with a different mean, one at a time. Each schedule remained in effect for 5,000 to 45,000 generations, or time steps, after which the next schedule was arranged, and so on. Each schedule yielded an average rate of emission of the target operant, R, and an average rate of reinforcement, r.

At the beginning of the experiment, an initial interval from the RI schedule was started, and the organism emitted its first behavior according to the method described

earlier. If the emitted behavior came from the target class, and if its latency since the last reinforcement (or since the start of the session in this case) equaled or exceeded the scheduled random interval, then a reinforcer was delivered. A new generation of behaviors was then produced using a fitness density function, and a new interval from the RI schedule was started. The organism then emitted its second behavior, and so on. If at any time a target operant was emitted but not reinforced, or if the emitted behavior did not come from the target class, then a new generation of behavior was produced from random parents, after which the organism emitted its next behavior, and so on.

3.1 Parametric Study of the Form and Mean of the Fitness Density Function

Five experiments were conducted using a uniform fitness density function, five were conducted using a linear fitness density function, and five were conducted using an exponential fitness density function. The five experiments for each function form arranged mean fitnesses (μ in Equations 2-4) ranging from 10 to 200. In all experiments the midpoint fitness definition and bitwise reproduction method were used. Gaussian mutation with a standard deviation of 25 was used to produce mutants of 3% of each generation's behaviors.

The behavior of the artificial organism in these experiments reached a dynamic equilibrium with the RI schedule such that the momentary rate of the organism's behavior varied around a stationary mean value. Reinforcements tended to pull the organism's population of bit strings into the target class, while nonreinforcement tended to pull the population of bit strings out of the target class and return the organism to its baseline state.

An example of the steady-state behavior of the artificial organism over the range of RI schedules used in these experiments is shown in Figure 1. The data in the left panel were generated using a linear fitness density function with $\mu = 40$. Data are shown from only the last 500-generation block, and for only a few of the RI schedules used in the experiment. The smooth curve is the best fitting hyperbola (Equation 1), which accounts for 98% of the variance in the target behavior. The outcome shown in this panel is typical of experiments with live organisms. It also may be worth noting that the method of arranging RI schedules and of averaging response and reinforcement rates used in these experiments are identical to the methods used in experiments with live organisms. Data in the right panel of Figure 1 were generated using an exponential fitness density function with $\mu = 40$. The data were averaged over approximately 40 500-generation blocks and are shown for all the RI schedules used in the experiment. The smooth curve is the best fitting hyperbola, which accounts for more than 99% of the variance in the target behavior.

The outcome shown in the right panel of Figure 1 is typical of the outcomes of all experiments in this series. When the data were averaged over 5,000 to 45,000 generations, which reduced the standard errors of these means to very small values, Equation 1 accounted for essentially all the variance (> 99% in most cases) in the artificial organism's target behavior, and this was the case regardless of the form or mean of the fitness density function used to generate the data.

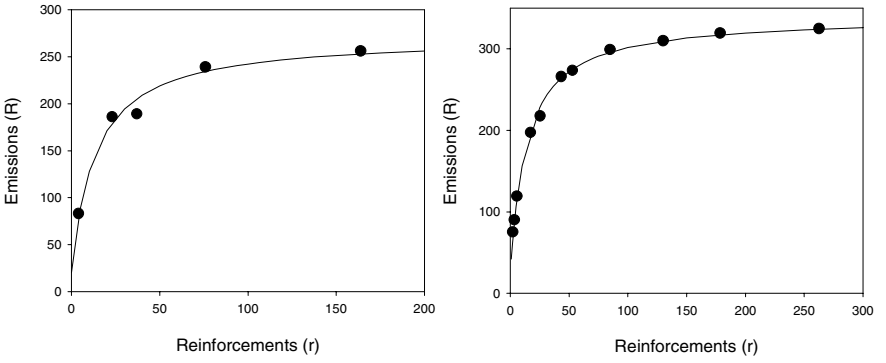


Fig. 1. Target behavior emissions per 500-generation block plotted as a function of contingent reinforcements per 500-generation block. Smooth curves are best fitting hyperbolas (Equation 1). The left panel, which shows data from only the last 500-generation block, is similar to the outcome of an experiment with a live organism. The right panel shows data for more RI schedules (using a different fitness density function), and averaged over about 40 500-generation blocks.

The hyperbolic form of the function relating the rate of the target behavior and the rate of reinforcement obtained for the target behavior was further tested by comparing it to fits of a two-parameter asymptotic exponential, a two-parameter asymptotic power function, and a two-parameter ramp function. The latter is a piecewise continuous function consisting of a line that increases from the origin, followed by a constant value that begins at some positive reinforcement rate. This is arguably the simplest function form that can describe data that ascend from the origin and then level off. The asymptotic exponential and asymptotic power functions have differential properties similar to those of a hyperbola.

The four function forms (including the hyperbola) were compared on the basis of the percentage of variance they accounted for, and in terms of the randomness of the residuals left by their least squares fits. Based on these criteria, the hyperbola provided a better fit to the data from the fifteen experiments in this series than did the other function forms and it accounted for essentially all the variance in the data. The other forms accounted for significantly less variance, and left residuals that showed significantly more deviations from randomness. These results indicate that the artificial organism's steady-state behavior was consistent with the quantitative law of effect, and that the hyperbolic form of the relationship between target behavior frequency and reinforcement frequency was both unique and robust, that is, it provided a better description of the data than other, similar, function forms, and did not depend on the form or mean of the fitness density function.

3.2 Study of Variations in the Components of the Evolutionary Algorithm

In a series of twelve experiments, different combinations of fitness definition, reproduction method, and mutation method, along with various fitness density function forms, were studied. Specific component variations tested included the specific individual fitness definition, crossover reproduction, bit-flip mutation, and random

individual mutation. Twelve combinations of these component variations, along with the variations used in the first series of experiments were tested.

Least squares fits of a hyperbola to the data from these experiments accounted for essentially all the variance of the target behavior. The three alternative function forms were also fitted to the data and were found to account for significantly less variance than the hyperbola, and to leave residuals showing significantly more deviations from randomness. These results indicate that the hyperbolic relationship between target behavior frequency and reinforcement frequency did not depend on any specific definition of fitness, or on any specific implementation of reproduction or mutation, or on any specific combination of these component variations, although only a subset of the possible combinations of component variations was tested.

3.3 Parametric Study of Mutation Rate

Using the same component variations as in the first series of experiments, together with a linear fitness density function, all possible combinations of five fitness function means (10, 20, 40, 100, and 200) and six mutation rates (1%, 3%, 5%, 10%, and 20%) were studied in thirty experiments.

Again, least squares fits of a hyperbola accounted for essentially all the variance in the target behavior for these thirty data sets, and the three alternative function forms accounted for significantly less variance and left residuals that were significantly less random than the hyperbola. These results indicate that the hyperbolic form of the behavior-reinforcement relationship does not depend on the mutation rate.

Data from these experiments also permit a parametric examination of the effects of mean parental fitness and mutation rate on the parameters, k and r_e, of the hyperbola. Both were affected by the two variables, but k was much more strongly affected by mean parental fitness, whereas r_e was much more strongly affected by mutation rate. Recall that lower mean parental fitnesses cause fitter parents to be selected for mating. The results of these experiments showed that the fitter the parents selected for mating, the higher the asymptote of the hyperbola. Put another way, a given reinforcement rate, r, generated a higher response rate, R, the fitter the parents selected for mating. This effect is analogous to the effect of reinforcer magnitude on the behavior of live organisms. Hence larger reinforcer magnitudes can be represented by lower mean parental fitnesses in the evolutionary algorithm.

The principal effect of higher mutation rates was to increase the value of r_e and hence decrease the curvature of the hyperbola. Put another way, to achieve a given response rate, R, a greater reinforcement rate, r, was required the greater the mutation rate. Not surprisingly, then, mutation diluted the effect of reinforcement.

4 Conclusion and Future Directions

An evolutionary algorithm, the operation of which was not associated with an optimality condition, was used as a behavioral mechanism for an artificial organism, and was shown to generate steady-state behavior consistent with the well-established quantitative law of effect (Equation 1). Three series of experiments demonstrated that this result was robust and unique. The result was robust inasmuch as it was independ-

ent of the specific methods of implementing the rules of the evolutionary algorithm. Evidently, robust outcomes of evolutionary algorithms for reinforcement learning are not unusual [8]. The result was unique inasmuch as a hyperbola described the organism's steady-state behavior better than other, similar, function forms.

While steady-state behavior was the focus of this research, the evolutionary algorithm used in these experiments also gives an artificial organism the ability to adapt continuously to a dynamic environment by tracking changes in reinforcement contingencies. In the absence of reinforcement, the organism's repertoire reverts to its baseline state over a number of generations. The extent to which the dynamics of the evolutionary algorithm, such as its time course, correspond to the dynamics of the behavior of live organisms remains a topic for future research.

The state space of the artificial organism used in these experiments was very simple, which reflects its origin as an analog of the basic unit of behavioral experimentation, namely, a single organism in a restricted environment that interacts with only one class of behavior. Indeed, the artificial organism operated in just one state, from which it could emit one of only two classes of behavior. This restricted repertoire is much simpler than the policies that are often studied in research on utility-based and action-based reinforcement learning [5, 8]. But just as the basic laboratory preparation is a building block for more complicated experimental environments, the evolutionary algorithm described here might prove useful as a building block for dealing with more complicated state spaces.

In the experimental analysis of behavior, a state space is characterized by what is called a discriminative stimulus, and behavior associated with that (often complex) stimulus is said to be under its control or, more generally, under stimulus control. Mapping behavior to discriminative stimuli in behavior analysis is analogous to mapping actions to states in artificial intelligence research, although the former mapping is always probabilistic. A sequence of mappings between discriminative stimuli and behavior constitutes what would be referred to in artificial intelligence research as a policy. The work described in this article dealt with a single mapping of one state to a set (with only two members) of probabilistic actions. There are many approaches to building a more complicated policy. One is to switch from 10-character bit strings to 100-character integer strings, each of which represents the artificial organism's behavioral repertoire in the presence of a different discriminative stimulus. The repertoire represented by each integer string would evolve (presumably in conformance with Equation 1) in the presence of its discriminative stimulus, and the collection of integer strings at any moment would constitute the organism's policy at that moment. This approach would engage the problem of credit assignment inasmuch as reinforcement could be delivered after a sequence of actions. Methods of dealing with this problem include using chaining mechanisms and conditioned reinforcement, an approach taken by Touretzky and Saksida [13], or using a delay-of-reinforcement gradient that is informed by findings from live organisms.

This research lies at the interface of the experimental analysis of behavior and artificial life. The evolutionary algorithm described in this article is a good candidate for a dynamics of live behavior, and it might be a useful building block for more complex artificial organisms that have the ability to adapt continuously to complex environments.

References

1. Davison, M., McCarthy, D. *The matching law*. Erlbaum, Hillsdale, N.J. (1988)
2. Daw, N.D., Touretzky, D.S. Operant behavior suggests attentional gating of dopamine system inputs. *Neurocomputing* 38-40 (2001) 1161-1167.
3. Herrnstein, R.J. Relative and absolute strength of response as a function of frequency of reinforcement. *Journal of the Experimental Analysis of Behavior*, 4 (1961) 267-272.
4. Herrnstein, R.J. On the law of effect. *Journal of the Experimental Analysis of Behavior*, 13 (1970) 243-266.
5. Kaelbling, L.P., Littman, M.L., Moore, A.W. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4 (1996) 237-285.
6. McDowell, J. J A computational model of selection by consequences. *Journal of the Experimental Analysis of Behavior* 81 (2004) 297-317.
7. McDowell, J. J, Bass, R., Kessel, R. A new understanding of the foundation of linear system theory and an extension to nonlinear cases. *Psychological Review* 100 (1993) 407-419.
8. Moriarty, D.E., Schultz, A.C., Grefenstette, J. J Evolutionary algorithms for reinforcement learning. *Journal of Artifical Intelligence Research* 11 (1999) 241-276.
9. Rachlin, H., Battalio, R., Kagel, J., Green, L. Maximization theory in behavioral psychology. *Behavioral and Brain Sciences* 4 (1981) 371-417.
10. Seth, A.K. Evolving behavioural choice: An investigation into Herrnstein's matching law. In Floreano, D., Nicoud, J.D., Mondana, F. (eds.) *Proceedings of the Fifth European Conference on Artifical Life*. Springer-Verlag, Berlin Heidelberg New York (1999) 225-236.
11. Seth, A. K. Modeling group foraging: Individual suboptimality, interference, and a kind of matching. *Adaptive Behavior*, 9 (2002) 67-90.
12. Sutton, R.S., Barto, A.G. *Reinforcement learning: An introduc*tion. MIT Press, Cambridge, MA (1998).
13. Touretzky, D.S., Saksida, L.M. Operant conditioning in Skinnerbots. *Adaptive Behavior*, 5 (1997) 219-247.

Self-adaptation of Genome Size in Artificial Organisms

C. Knibbe[1], G. Beslon[1], V. Lefort[1], F. Chaudier[2], and J.-M. Fayard[3]

[1] Prisma lab., INSA Lyon, 69621 Villeurbanne Cedex, France
[2] Biosciences Department, INSA Lyon, 69621 Villeurbanne Cedex, France
[3] BF2I - UMR 0203, NRA/INSA Lyon, 69621 Villeurbanne Cedex, France
guillaume.beslon@insa-lyon.fr

Abstract. In this paper we investigate the evolutionary pressures influencing genome size in artificial organisms. These were designed with three organisation levels (genome, proteome, phenotype) and are submitted to local mutations as well as rearrangements of the genomic structure. Experiments with various per-locus mutation rates show that the genome size always stabilises, although the fitness computation does not penalise genome length. The equilibrium value is closely dependent on the mutational pressure, resulting in a constant genome-wide mutation rate and a constant average impact of rearrangements. Genome size therefore self-adapts to the variation intensity, reflecting a balance between at least two pressures: evolving more and more complex functions with more and more genes, and preserving genome robustness by keeping it small.

1 Introduction

As Maynard-Smith pointed out in 1982, the evolution of large-scale genomic features is "one of the most difficult, perhaps *the* most difficult, question in evolutionary biology" [1]. Since then, molecular biology provided us with huge data about individual genes. Still, little is known about the forces that shape the global structure of the inheritable information in living systems. Experimental evolution of natural systems like cultivable and fast-replicating bacteria takes years [2]. Artificial organisms, by allowing for rapid experiments and parameter control, can help understanding the basic processes at work in evolving systems.

Although genetic algorithms proved useful to study population-level problematics, they cannot capture the genome dynamics. Their genotype-phenotype map, where the contribution of each gene relies on its locus, requires both gene number and gene order to be predefined. This forbids changes in genome length and leads to a frozen gene organisation. Pioneering work aiming at removing these constraints [3,4] kept a fixed phenotypic structure, with a given number of functions, each of them having to be performed by one gene. They therefore needed an external daemon to choose the expressed genes.

Natural genomes owe their degrees of freedom to a flexible phenotypic structure and to the existence of an intermediate level between the genotype and the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 423–432, 2005.

phenotype: the set of proteins, whose interactions ensure the survival and reproduction functions. Therefore, to study the evolution of the genomic structure, we introduced such a level into artificial organisms competing for reproduction. Each of them owns a genome encoding basic functional elements, whose interactions produce the phenotype. Both the genomic and the functional structures are evolvable, by the means of punctual mutations and large-scale rearrangements of the genetic material.

Section 2 presents this platform, called *aevol*, notably detailing how it enables us to test self-adaptation hypothesis. In section 3, we focus on the experiments we carried out to test the influence of the mutational pressure on genome size, and we discuss these results in section 4. We conclude in section 5.

2 Designing Organisms with Flexible Genomic and Functional Structures

The *aevol* system aims at giving as much freedom as possible to let the different levels self-organise. To reach this objective, some features of natural genetic systems were reproduced: (i) the genome is made up of a variable number of genes separated by non-coding sequences, (ii) mutations can modify the genomic structure, (iii) the expression level and the function of a gene are not predefined, and they do not depend on its position but on the local sequence, and (iv) the phenotype results from the interactions of basic functional elements encoded by genes. The remainder of this section describes how these features are implemented.

2.1 From Genotype to Phenotype

As shown by Figure 1, the genome is a circular, double-strand binary string, where 0 and 1 are the complementary bases. Not all the positions are functional: coding sequences (genes) are detected by transcription-translation process inspired by bacterial genetics. Genes are then translated into basic functional elements (proteins). These elementary functions are combined together to get the global abilities of the organism (phenotype).

Transcription. Sequences called promoters and terminators define both the boundaries and the expression level of the transcribed regions. A preliminary study showed that long *and* frequent terminators associated with rare promoters (i) allow for the emergence of coding sequences, and (ii) limit the overlaps of transcribed regions, thereby giving more freedom for gene rearrangements. Thus, a long consensus sequence was defined to detect promoters (sequences whose Hamming distance with the consensus is $d \leq d_{max}$), whereas terminators are located using their secondary structure ($abcd***\bar{d}\bar{c}\bar{b}\bar{a}$). The expression l level of a transcribed region depends on the similarity between its promoter and the

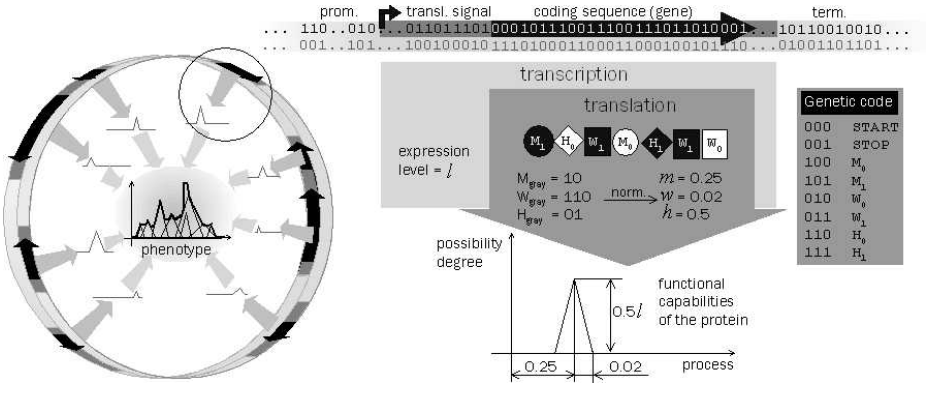


Fig. 1. Overview of the phenotype computation

consensus [1]: $l = 1 - \frac{d}{d_{max}+1}$. The following experiments were carried out with a consensus of 28 base pairs (bp) and $d_{max} = 4$.

Translation. During the translation phase, transcribed regions are searched for coding sequences: once a start signal is found, the subsequent positions are read three by three (codon by codon) until a stop signal is reached. The start signal is made up of a Shine-Dalgarno-like sequence followed by the START codon (011011***000), and the stop signal is simply the STOP codon (001). Indeed, a longer start signal limits the overlaps between coding sequences, and hence the rigidity of the gene organisation. Once the coding sequences are located, an artificial genetic code is used to translate them into proteins, able to either activate or inhibit processes.

These functional capabilities are expressed within a fuzzy logic framework: the set of processes that can be achieved in our artificial world is an interval of $\mathbb{R}$ ([0, 1] here), and each protein is represented by the fuzzy subset of processes it is involved in. This fuzzy formalism enables us to assign a non-null *possibility degree* to each process the protein inhibits or contributes to. The action of a protein can therefore be described by its bell-shaped possibility distribution, approximated by piecewise linear distributions (Figure 1).

Three parameters are necessary to describe such a distribution: its mean m, its width w, and its maximal height H. The main process m and the interaction potential w are supposed to depend on the coding sequence only. But the maximal possibility degree H is limited both by the intrinsic efficiency h of the protein and by the expression level l of the region. The genetic code enables us to assign the contribution of each codon to the value of m, w or h, via a Gray

[1] This simplistic notion of protein quantity was not introduced to model the complex regulation processes at work in living organisms, but rather to allow new gene copies to reduce temporarily their phenotypic contribution, thereby allowing their sequences to diverge.

encoding (Figure 1). The sign of h determines the activator or inhibitor nature of the protein, and its absolute value is used to compute $H = l|h|$.

Functional Interactions. A given process may be achieved by several proteins and inhibited by several others. Therefore, the fuzzy set of processes the organism is able to perform contains the processes that are activated AND NOT inhibited by its proteins. If A_i is the set of the i-th activator protein and I_j the set of the j-th inhibitor protein, the set of the organism is $P = (\cup_i A_i) \cap \overline{(\cup_j I_j)}$, and its phenotype is the possibility distribution of P. Lukasiewicz fuzzy operators are used to perform this combination.

2.2 Selection

The environment is also represented by a fuzzy subset of processes, whose possibility distribution is arbitrarily defined. An organism is well adapted if it performs the processes that are feasible in the environment. The higher the gap $g = \int_0^1 |E(x) - P(x)|dx$ between the environmental distribution E and the phenotype P of an organism, the lower its offspring size. The environmental distribution E we used for the following experiments is shown by Figure 2(b).

The population management is similar to the classical methods used in genetic algorithms. The population size is fixed, and at each time step, all parents are replaced by the offspring. The fitnesses are assigned by linear ranking of g. The actual selection is done by stochastic sampling with replacement.

2.3 Variation Operators

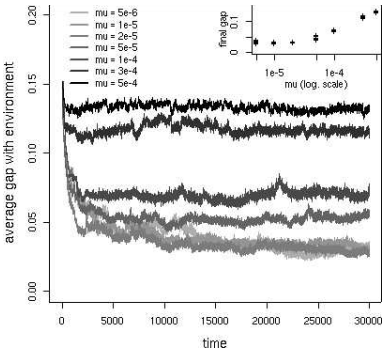
The genome of each selected organism is replicated with eventual random errors, affecting a few positions (local mutations) or huge genomic segments (rearrangements), regardless of their function. Genetic exchange between organisms (crossover) is also implemented, but was not used in the following experiments.

Three types of local mutations can be performed at a given position: "switching" its value, inserting or deleting one to six bp. For each mutation type, the number of events per replication follows the binomial law $\mathcal{B}(L, \mu)$, where L is the genome length and μ the per-locus mutation rate.

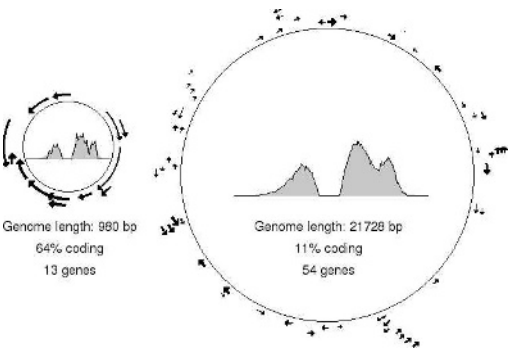
Large-scale rearrangements involve the choice of a genomic segment to be deleted, duplicated, translocated (moved) or inverted. The numbers of events per replication also follow binomial laws. The boundaries of the segment and the eventual insertion point are chosen with a uniform law, edge effects being avoided by the circularity of the genome.

2.4 Properties of the System

The proteome level we introduced removes the rigidity of the functional structure: a given process may be achieved by a variable number of functional elements. This in turn removes the rigidity of the genomic organisation; the genome



(a) Evolution of the average gap $\bar{g}$ between the phenotypes and the environmental distribution, for various per-locus mutation rates (μ).



(b) Best organism after 30000 generations, obtained with $\mu = 10^{-4}$ (left) or $\mu = 5.10^{-6}$ (right) for each mutation type. The arrows represent the genes, the black curve the phenotype, and the grey area the environmental distribution E.

Fig. 2. While organisms adapt to their environment, a specific genomic organisation emerges, depending on the mutation rates

can undergo rearrangements and indels without preventing phenotype computation. Genome length, gene number and gene order are therefore free to change.

This property enables us to test evolutionary hypothesis involving genome self-organisation through and according to the selection and the variation mechanisms. Indeed, organisms are selected on the basis on their phenotype only, independently of their genomic features: different genome lengths or different gene orders can give the same phenotype, and hence the same fitness. Nevertheless, while the organisms adapt to their environment (Figure 2(a)), some reproducible genomic and functional structures emerge in the long term, for each parameter set. Figure 2(b) shows two different genomic structures, obtained with the same selection method but with different mutation rates.

3 Experimenting the Influence of the Mutational Pressure on Genome Size

We used this system to investigate specifically the evolutive pressures acting on genome size. In the experiments we carried out, the seven possible mutations had the same per-locus rate μ, ranging from 5.10^{-6} to 5.10^{-4} mutations per bp. For each mutation rate, we let five asexual populations of 1000 artificial organisms evolve independently during 30000 generations, within the steady environment shown in Figure 2(b). The populations were seeded with random genomes of 5000 bp.

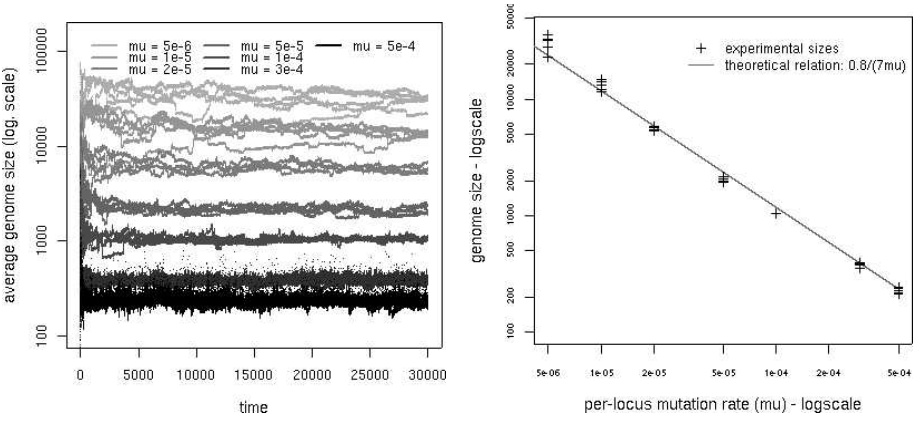


Fig. 3. Genome size stabilises at a value that depends on the mutation rate μ

3.1 Stability of Genome Size, Convergence on Local Optima

The initial genomes generally do not contain any gene, but after a few generations, local mutations allow a first gene to be expressed. If the set of processes it achieves are feasible in the environment, this first gene is maintained and quickly duplicated. The actions of all copies are summed up, and the organism's abilities eventually exceed the environment's, which is deleterious in our model. Some of the copies are subsequently lost, while other copies diverge: local changes in their coding sequences modify the average process m they are involved in, allowing the corresponding proteins to move along the functional axis and achieve new processes.

To close the gap with the environmental distribution, the organisms could then adapt the efficiency or the expression level of the genes they already own. Yet a finer tuning could be achieved by acquiring more and more balancing inhibitor/activator genes. However, Figure 3 shows that after a short phase of massive gene acquisition, the genome size reaches an equilibrium, and so the fitness does, trapping the populations on local optima (Figure 2(a)).

When the phenotype is close to the environmental distribution, duplicating a gene becomes more deleterious, which undoubtedly slows down the gene acquisition. Indeed, after the first phase of massive gene acquisition by duplications, the fixation rates of both duplications and large deletions drop to 0.01-0.02 events per generation, whereas other mutation types all stabilise at a higher value, up to 0.15 events per generation, depending on the mutational pressure μ. However, it is still theoretically possible to create a gene "from scratch" or to duplicate a coding sequence without its promoter and letting it diverge before expressing it. Therefore, the genome should continue to grow, although more and more slowly.

Moreover, the deleterious effect of the duplications cannot explain simply that the higher the mutation rates, the smaller the equilibrium genome size, and the lower the average fitness: high mutation rates prevent the organisms

from developing complex, highly adapted functions. Therefore, surprisingly, being trapped on local optima is not the consequence of too low a mutation rate. On the contrary, it seems to come from the global pressure of the mutation events on the genome structure, as we shall argue in the following sections.

3.2 Convergence Towards a Constant Genome-Wide Error Rate

If the genome size L varies, then the expected number of mutations per replication $M = 7\mu L$ changes too. Now, while a low M prevents the exploration of new solutions, a high M endangers the robustness of the current one. The existence of a genomic mutation rate, named *error threshold*, "beyond which structures created by an evolutionary process are destroyed more frequently than selection can reproduce them" [5] was demonstrated both in quasi-species models [6] and in genetic algorithms [5]. However, for both models, the genome size is generally fixed, and the mutation rate must be carefully chosen to balance efficiently exploitation and exploration.

Figure 4(a) shows that in our experiments, where genome size is free to vary, the equilibrium size is such that M takes the same value (around 0.8 mutations per replication), regardless of the mutational pressure μ. The equilibrium size can therefore be predicted from μ with an hyperbolic relation (Figure 3): $L \simeq \frac{0.8}{7\mu}$.

Thus, the genome size stability reflects a compromise between - at least - two contradictory pressures: on the one hand, improving the phenotype with more and more genes, and on the other hand, resisting mutational pressure by keeping genome size small.

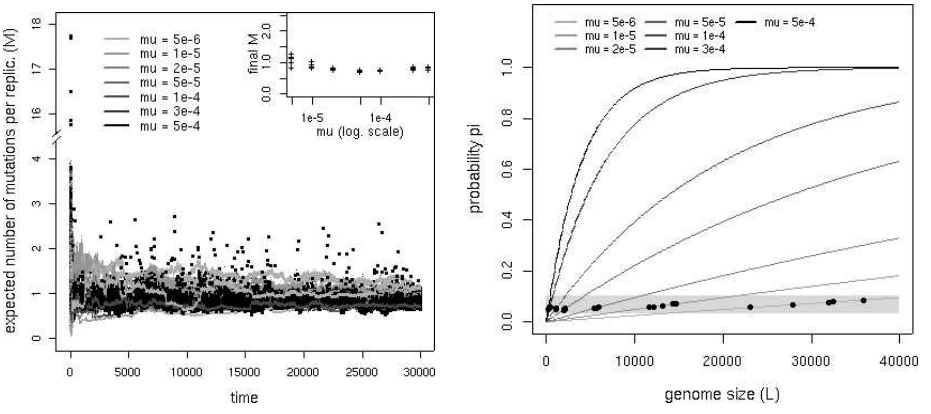
3.3 Convergence Towards a Constant Impact of Rearrangements

The probability that during a replication, a given position α is affected by a local mutation is the same (μ) whatever the genome size. On the contrary, the average impact of a rearrangement does increase with genome length L. Indeed, for a given rearrangement type, say inversion, this impact can be estimated by the probability π for α to be affected by at least one of the inversions performed during a replication. This probability can be approximated by $\pi = 1 - \left(1 - \mu\frac{L+1}{2L}\right)^L$ under the hypothesis that the successive rearrangements are independent.

Figure 4(b) shows that in our experiments the genome size spontaneously converges towards the same value of π (around 0.05): the genome size is such that the rearrangement phase of the replication keeps the same average impact when μ changes. Besides, figure 4(b) shows that the lower μ is, the weaker the slope of π is, which could explain the high run-to-run variability in genome size observed for the low mutation rates (Figure 3).

3.4 Changing Mutation Rates During the Evolution

To confirm these results, and to disentangle the effects of small mutations and rearrangements, we carried out additional experiments: we let a population of



(a) The genome-wide error rate M quickly stabilises at the same value, whatever the per-locus error rate.

(b) Theoretical relation between π and L. Black points locate the experimental genome sizes.

Fig. 4. A need for robustness could explain the limitation of genome size

artificial organisms evolve during 30000 generations with $\mu = 10^{-5}$. Then we changed the rate of the small mutations and/or the rate of the rearrangements, and let the evolution go on.

When all mutation rates are increased, the acquired genomic structure is quickly displaced by a new, shorter one, more robust but less fit. This shows that the shrinkage effect can be surprisingly strong, compared to the pressure for individual adaptation. When, on the contrary, all mutation rates are lowered, this constraint relaxes, enabling the genome to grow. In both cases, final genome sizes can be predicted with the relations $M = 0.8$ or $\pi = 0.05$.

Increasing only the rearrangement rates suffices to make the genome shrink. On the contrary, if we keep high rates for local mutations and reduce rearrangement rates, genome size does not increase. Low rates for both small and large-scale mutations are thus required to make it grow. Analysing the effects of the local mutations, in relation with the coding proportion of the genome, should help us understanding this asymmetric behaviour.

4 Discussion

To explain both the diversity and the stability of genome sizes observed in natural organisms, a mutational equilibrium model was recently discussed [7]. This model relies on two different bias: on the one hand, a mutational bias towards small deletions in the indel mechanisms, and on the other hand, a higher fixation rate of large insertions compared to large deletions. Although such bias can exist in natural species [8], our experiments show that they are not mandatory to stabilise genome size.

Other hypotheses accounting for the steady genome sizes of natural organisms involve natural selection acting directly at the genomic level, either as a stabilising force maintaining the DNA content at a physiological optimum [9], or as a directional pressure counterbalancing the proliferation of "selfish" or "junk" DNA [10,11] by favouring a short replication time. However, it was shown that there is no correlation between genome size and doubling time among prokaryotes [8]. Besides, in our organisms, genome size does not increase infinitely, although it has no effect on the reproduction rate.

Natural selection can also limit genomic growth a posteriori because of mutational load effects. As mentioned above, the larger a genome is, the more errors occur during its replication. Now quasi-species theory predicts that natural selection favours the set of genotypes, linked by mutation, whose average fitness is highest [12]. It was shown that an evolving population concentrates on the most robust genotypes of the neutral network of high fitness [13]. Experiments with the Avida platform [14] confirmed that digital organisms occupying low but flat fitness peaks can even displace fitter but less robust ones, provided that mutation rates are high enough [15].

In both studies, the genome length was fixed, but what happens if several genotypes with different lengths but the same fitness are in competition? Smaller genomes will undergo less mutations per replication, thus the size of the steps on their peaks will be smaller: everything happens as if they stood on a flatter peak than the larger genomes. Quasi-species theory would therefore predict that under high mutation rates, these smaller genomes will win the competition. This is indeed what our experiments tend to show.

Thus, under high mutation rates, the long-term selection for robustness is clear, quite unlike the genomic growth under low mutation rates. In our system, there is undoubtedly a pressure to evolve more complex functions involving more genes. But longer genomes also undergo more mutations per replication, which can compensate for a low mutation rate and allow for the exploration of new parts of the fitness landscape. A long-term selection for evolvability could therefore also occur.

5 Conclusion and Future Work

Experiments with our artificial system confirmed the intuitive idea that genomic growth, leading to more and more complex phenotypes, can be efficiently limited by a need for robustness. Rearrangements, and especially duplications and deletions, seem to play a key role in this equilibrium: they are the agents of genome length variation, and at the same time, genome length seems to be limited by their average impact.

The selective pressures that actually make the genome grow towards this maximum value have to be investigated further, notably to test the existence of a selection for evolvability, and to understand the role of local mutations in genomic growth. A detailed study of the effects of each mutation type on the fitness, in relation with the gene number and the coding proportion of the genome,

should help us understanding this process. This study should also investigate the influence of the selection intensity on the robustness constraint.

Besides, our system also enables us to study the evolution of gene organisation: since the functional structure and the genomic structure co-evolve, we can analyse the putative retro-actions of the functional level on the gene organisation, notably those leading to genetic modularity.

References

1. Maynard-Smith, J.: Overview - unsolved evolutionary problems. In Dover, G.A., Flavell, R.B., eds.: Genome evolution, New York, NY, Academic Press (1982) 375–382
2. Lenski, R.E.: Phenotypic and genomic evolution during a 20,000-generation experiment with the bacterium *Escherichia coli*. Plant Breeding Reviews **24** (2004) 225–265
3. Goldberg, D.E., Deb, K., Kargupta, H., Harik, G.: Rapid accurate optimization of difficult problems using fast messy genetic algorithms. In Forrest, S., ed.: Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, CA, Morgan Kaufmann (1993) 56–64
4. Burke, D.S., De Jong, K.A., Grefenstette, J.J., Ramsey, C.L., Wu, A.S.: Putting more genetics into genetic algorithms. Evolutionary Computation **6** (1998) 387–410
5. Ochoa, G., Harvey, I., Buxton, H.: Optimal mutation rates ans selection pressure in genetic algorithms. In: Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2000), San Francisco, CA, Morgan Kaufmann (2000)
6. Eigen, M., Schuster, P.: The hypercycle: A principle of natural self-organization. Springer-Verlag (1979)
7. Petrov, D.A.: Mutational equilibrium model of genome size evolution. Theoretical Population Biology **61** (2002) 533–546
8. Mira, A., Ochman, H., Moran, N.A.: Deletional bias and the evolution of bacterial genomes. Trends in Genetics **17** (2001) 589–596
9. Gregory, T.R.: Coincidence, coevolution, or causation? DNA content, cell size, and the C-value enigma. Biological Reviews of the Cambridge Philosophical Society **76** (2001) 65–101
10. Doolittle, W.F., Sapienza, C.: Selfish genes,the phenotype paradigm and genome evolution. Nature **284** (1980) 601–603
11. Ohno, S.: So much junk DNA in our genome. In Smith, H.H., ed.: Evolution of Genetic Systems, New York, Gordon and Breach (1972) 336–370
12. Eigen, M., McCaskill, J., Schuster, P.: The molecular quasi-species. Adv. Chem. Phys. **75** (1989) 149–263
13. Van Nimwegen, E., Crutchfield, J.P., Huynen, M.: Neutral evolution of mutational robustness. Proc. Natl. Acad. Sci. USA **96** (1999) 9716–9720
14. Ofria, C., Wilke, C.: Avida: A software platform for research in computational evolutionary biology. Artificial Life **10** (2004) 191–229
15. Wilke, C.O., Wang, J.L., Ofria, C., Lenski, R.E., Adami, C.: Evolution of digital organisms at high mutation rates leads to the survival of the flattest. Nature **412** (2001) 331–333

An Architecture for Modelling Emergence in CA-Like Systems

Fiona Polack, Susan Stepney, Heather Turner,
Peter Welch*, and Fred Barnes*

Department of Computer Science, University of York,
Heslington, York, YO10 5DD, UK
*Computing Laboratory, University of Kent,
Canterbury, CT2 7NF, UK
{fiona, susan, turner}@cs.york.ac.uk
{p.h.welch, f.r.m.barnes}@kent.ac.uk

Abstract. We consider models of emergence, adding downward causation to conventional models where causation permeates from low-level elements to high-level behaviour. We describe an architecture and prototype simulation medium for tagging and modelling emergent features in CA-like systems. This is part of ongoing work on engineering emergence.

Keyword: Cellular Automata, emergence, occam-pi , simulation.

1 Introduction

This paper represents part of ongoing research to establish engineering principles for complex emergent systems. Various systems require several levels of description; for example, the behavioural descriptions of individual components and of some aggregate. In an *emergent system*, there is a discontinuity in the descriptions of these various layers. For example, the low-level components might be described as changing state, whereas the system description might be in terms of the movement of patterns. The upper, system, level describes the required emergent properties.

We consider complex emergent systems, comprising many simple components. Often-cited examples of complex emergent systems include network navigation by ants (real or simulated), construction by termites, swarming and flocking, for example by birds or their simulated equivalent, boids, and cellular automata (CAs).

Engineering is a quality-enhancing activity, and is essential for the safe exploitation of emergence in nature-inspired computational systems; the engineered emergent system would be robust, with assurance of functionality and safety. In exploring emergent systems engineering, we are looking at compositionality and refinement. We start with simple emergent systems, specifically CAs, and derive more general guidance from our observations.

Elsewhere [9], we describe a system architecture to underpin engineering of complex emergent systems. We identify *three key elements*: the high-level description of the required system; the specification of the components that form

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 433–442, 2005.

the lowest level of the system; and the specification of the representation that integrates the first two elements. Conventional development approaches, relying on a linear reduction in non-determinism (data and process refinement; model-driven development, etc) are applicable within each element, but the low-level system components cannot be systematically derived from the system specification. The components are fundamentally different from the overall system, and cannot be described using the same language concepts.

In this paper, we explore extraction of a layered component model. The introduced layer maps from the component language towards the concepts of the emergent system. The layering approach explored here is derived from pure CA models; we deduce some characteristics of causal linkage among the system elements. We consider a system requiring emergence of specific gliders, and a case study simulating blood platelets.

2 Cellular Automata and Upward Causation

In a simple CA, such as Conway's Game of Life (GoL) [6], cell update rules and initial cell states completely determine the evolution of the CA. Emergence is detected when each cell state has a visual representation, and the repeated synchronous update of the cells reveals recognisable structures in space and time. When seeking to engineer emergence on such a CA, the three architectural elements are as follows.

1. Required emergent structures, such as gliders, described using relative motion concepts.
2. The CA, comprising many identical cell instances.
3. The representation, discretised space, to define cell neighbourhoods, and on which relative motion can be detected.

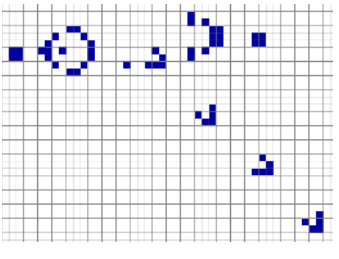


Fig. 1. A 2-D GoL Glider Gun

The CA has an *upward causal relationship* to the required emergence. For example, the upper part of figure 1 shows a GoL glider gun. In this part of the representation, we observe seemingly-random continuously-changing patterns. From the gun, a stream of gliders emerges, moving at a constant velocity, at 45 degrees from the vertical, down the screen. The glider gun is a simple result of applying the GoL rules to cells arranged in a 2-D regular grid, with a suitable arrangement of initial cell states. The high-level description of the observed behaviour of gliders does not have any role in the evolution of the CA; the described higher level behaviours are caused by the lower-level actions.

To be able to engineer emergent systems from high-level requirements, we need a more flexible and realistic causal model. First, we introduce our research case study; we then use some of its models to explore causality further.

3 The Case Study: Artificial Blood Platelets

Our working case study, a platform for specification, simulation and other emergent engineering aspects, is a system of artificial platelets. The desired emergent property is the sealing of breaks (wounds) in a tube or vessel.

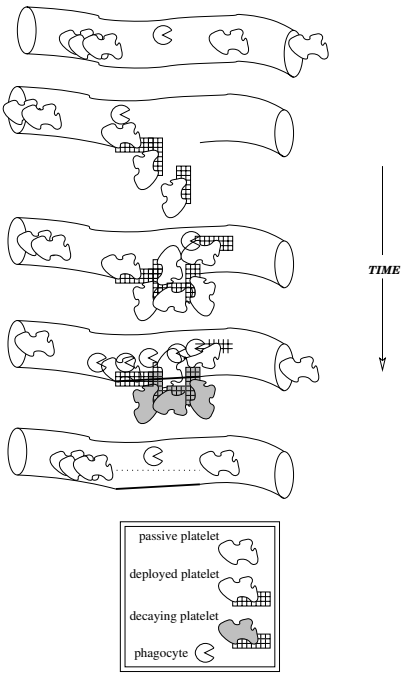


Fig. 2. Schematic of the artificial platelets

The model is loosely based on the medical process of *haemostasis*. Real platelets are passive quasi-cells carried in the bloodstream. A platelet becomes active when a balance of chemical suppressants and activators shifts in favour of activation, usually due to damage to cells or vessel linings. With sufficient stimulus, platelets become sticky and form clusters. This is the first phase in limiting blood-loss and healing a wound.

Our artificial platelet model, figure 2, assumes that the platelets can complete the entire wound-closing process. Our goal system might resemble Freitas' vision [5] of some 10^8 mechanical platelets of two microns diameter circulating in the blood, each carrying a fibre mesh. At a wound site, the mesh deploys, revealing sticky sections that bind other platelets and seal the wound; when the wound is healed, the mesh disperses.

In this paper, we consider the development of a model of platelet movement and clustering, the basis for a computer simulation. The high-level description is of platelets moved by blood flow through a vessel, with no independent means of locomotion. When platelets merge with other platelets, they form a slow-moving cluster. This description of platelet behaviour is at the same level of abstraction as the high-level glider description.

3.1 Upward Causation Model of Artificial Blood Platelets

Our first platelet model represents a blood vessel as a one-dimensional grid. Figure 3 shows eight time steps of a purpose-built CA running on this repre-

sentation, to simulate the flow of platelets, the formation of clusters, and the movement of clusters. Here, we see two clusters merging, and then free platelets joining the large cluster from behind.

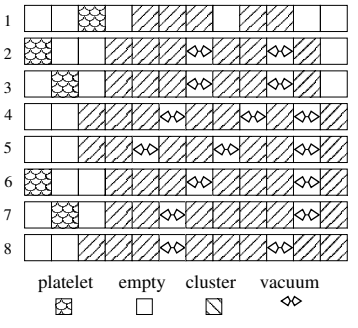


Fig. 3. 1D platelet CA

The CA rules are given in the Appendix. To achieve the required behaviour with pure CA rules, the first platelet in a cluster can move (non-deterministically), creating a vacuum; the cluster moves by successively passing the vacuum backwards. All free platelets move in each time step.

This is a pure CA, with a stochastic rule that determines whether a cluster moves in any step. There is only local communication, and only upward causation from the CA to the required emergent clusters.

4 Downward Causation and Rule Distribution

The simple CA platelet model is not ideal. The rate of movement of a cluster is very much slower than that of a free platelet, at most one cell per update, because cells cannot communicate throughout a cluster. Furthermore, a model where platelet locations control platelet movement by upward causality is not an adequate model of reality; we know that platelet aggregation influences the flow of blood and the flow of blood influences the aggregation of platelets. Such causal links are well-known in biological and other emergent systems [1].

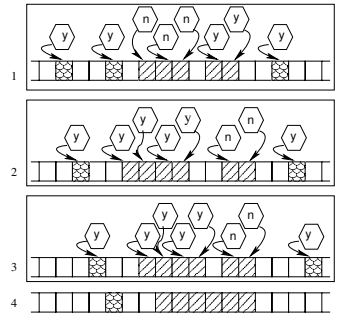


Fig. 4. Two-layer platelets

A revised model retains the one-dimensional CA structure at the lowest level, to simulate discrete physical locations that may contain a platelet. A more abstract level sits above the CA to model clustering characteristics. In this model, only the higher level "knows" that a particular platelet is part of a cluster. Figure 4 shows four time steps. The free platelets still move in every time step; each cluster either moves or stays put, depending on the decision made in the higher layer.

Conventional CA rules still determine whether a cell could change state, but the actual change requires permission to be communicated from the platelet layer, a *downward causation* from the higher level to the CA. The permission is used to co-ordinate the movement of platelets in a cluster, depending on the (non-deterministic) movement of the front platelet.

The higher layer can also be used to add further cluster behaviour, such as cluster breakdown and dispersion.

5 Discussion of Rule Migration

In the platelet models, the glider model [9], and other CA-based systems, the CA model has no inherent awareness of the structures that may emerge. In general, modelling is simplified if there are extra modelling layers that capture concepts expressed in the system-level language. Thus, the gliders are specified in terms of velocity not CA cell states; they can be identified by monitoring at a higher level, over many time steps. Similarly, platelet behaviour (here) is specified in terms of clustering characteristics; clusters are initially identified from the CA, but their persistent characteristics are modelled in the higher layer.

In the platelet model, the clustering rules have been taken out of the CA and migrated to the higher layer; the downward causation (permission) maintains integrity between the layers. We observe that, if the low-level rules are very compact, as in the GoL without the added requirement of structure identification, there may be no rule migration that makes the model simpler. The simplest additional layer provides "tagging", with no downward causation, for example as an aid to the detection of emergence. Thus, in the glider model, a higher level might be used to detect and highlight gliders; this is analogous to experiments in nature that use markers for tracking to collect data. At a level more akin to that of the platelet model, we might then wish to exercise control over, for example, what happens when two gliders collide; this would be accomplished via downward causation from the higher level to the relevant cells of the CA.

In engineering terms, the migrated rules are used to produce more natural, comprehensible models. Whether to choose the pure CA or a multi-level model is a *modelling decision*. The downward causation layers introduce control, and can be used to bring the power of the models closer to the full environmental interactions of real systems. In general, as the number of control aspects modelled in the abstracted layers increases, the behavioural similarities of the single-layer and multi-layer models become less apparent.

5.1 Emergence and Relativity

In physics, there is no absolute space; all motion is *relative*. It is also the case that all emergence is relative. Consider a single GoL glider; viewed from a sufficient distance, a glider moving across the screen is indistinguishable from a screen window being scrolled past a stationary glider. To perceive motion, there needs to be a frame of reference within the window. This could be a visible grid, a stationary (or slower-moving) CA structure, or other gliders. The glider is then seen to be moving *relative* to the other contents of the window.

In migrating CA rules upwards, we often move from an absolute to a relative perspective, taking the design nearer to the context in which the emergence is detectable. We would like to be able to abstract away from artificial representations, to use natural descriptions of these high-level rules. For example, when

modelling the layers on top of a CA, we should ignore the absolute grid representation, describing the emergence (gliders and other CA structures) abstractly, and without reference to lower level rules. We can then connect the high- and low-level elements by suitable causation links, to engineer the required emergence. The next section shows how a layered design can be implemented, preserving the relativity of the higher layer and the absolute lower-layer concepts.

6 An Implementation of the Layered Platelet Design

To explore simulations that demonstrate rule migration, we use a mobile extension of a traditional concurrent language. occam-π [1] is a small language that implements the communication strengths of Hoare's CSP[7] and the mobile aspects of Milner's π-calculus[8]. It takes the well-grounded semantics of these specification calculi, and provide a programming environment to support an engineering approach to the underlying mathematics[2,10].

The implementation of the platelet model uses occam-π static processes to represent the underlying CA, and mobile processes to model the activation and clustering of platelets. Downward causation is programmed as the mobile processes stimulating change in a CA cell. Upward causation is the reading of cell state by the mobile channels.

We can associate various visualisations to the simulation. An absolute-space model can be observed if static processes communicate their location and state to a display. A relative-location visualisation is achieved if mobile processes communicate their size and relative location to a display.

6.1 The occam-π Design

The occam-π model has a one-dimensional cell array, as before. Each cell is a static *server* process.

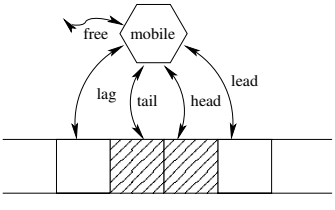


Fig. 5. A two-platelet cluster

One approach to simulating the two-layer platelet model it to associate a mobile process to each cell containing a platelet. The mobile process (figure 5) holds the size of the cluster and *client*-ends of four (multi-way) channels: `head` and `tail` connect to the first and last cells in a cluster; `lead` and `lag` connect to cells immediately ahead of and behind the cluster, acting as *feelers* to the cells round the cluster. The mobile process also holds the *server*-end of the `free` channel, used to merge adjacent clusters. The channel protocols allow two-way communication for the setting and retrieving of data and channel ends. Each cluster deposits the *client*-end of its `free` channel in the cell connected through `tail`. When the cluster moves, the `free` channel is thus dragged along the cells in turn.

[1] See http://frmb.org/kroc.html for the latest implementation.

When a platelet or cluster `head` enters the cell immediately behind another, the front mobile process (`m1`) detects its presence via an enquiry on the `lag` channel. The back process (`m2`) also detects that it is adjacent to another cluster via an enquiry on its `lead` channel, which points to the same cell as `m1`'s `tail`.

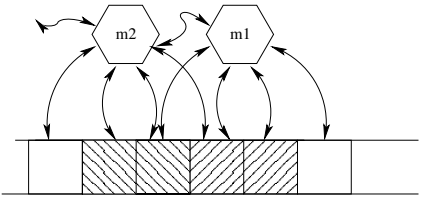


Fig. 6. Two coalescing clusters

Figure 6 shows the results of the `m2` process using its `lead` channel to acquire the client-end of `m1`'s `free` channel, resulting in direct communication between the two clusters. Once the link between the mobile processes is established, `m2` communicates its size, the client-ends of its `tail` and `lag`, and the server-end of its `free` channel to `m1`; `m2` then terminates. `m1` adds the received size to its own size, and overwrites its `tail`, `lag`, and `free` channel ends with those that it receives. `m1` has now assumed control of the combined cluster, shown in figure 7.

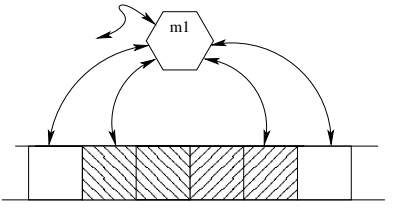


Fig. 7. The completed merger

Coordination of cluster merging and movement is safely and efficiently managed by *barrier synchronisation*. A two-phase cluster cycle is divided by barriers. Phase one detects when one cluster has other adjacent clusters (on one or both sides) and handles all the resultant cluster merges. In phase two, mobile processes determine the movement of their clusters. Barrier synchronisation in occam-π is extremely cheap (see [3]). All memory for terminated processes and discarded mobile channels is automatically freed (without garbage collection); there can be no memory leaks and the model runs indefinitely.

6.2 Extending the Platelet and Glider Simulations

The occam-π simulation allows us to explore the use of higher layers in a CA-based model, and to explore platelet simulation with additional control factors from environmental models. The mobile process tagging will also be used on a GoL simulation to facilitate automatic detection of incipient gliders. An element of downward causation could be added to the GoL, perhaps "clearing" the neighbours ahead of a new glider to prevent its being absorbed by background "noise".

The mobile processes used to tag gliders and link platelets implement relative location (i.e. connections to neighbouring cells); this information is held locally. Rules at the higher level refer only to relative properties, not absolute properties of the current grid location. The lowest level still uses an absolute grid, but this

is accessed only to display the result of each update cycle; individual cells are unaware of their absolute grid location.

7 Related Work

We are not alone in recognising that representations are often layered, such that point events at one level correspond in some approximate but definable way to actions with extent at lower levels. The point events are often invisible at higher levels. We are aware of at least three other research initiatives, in areas as diverse as model-driven architectures and real-time systems, which are discovering that layering is a key concept; no work has yet been published on these discoveries.

There are some similarities between our extra levels to control and interpret the CA behaviour and other CA-based research programmes; the difference is that others do not explicitly use their interpretation layers for downward causation. For example, in Fredkin's digital philosophy, readings of various parameters at various of six defined phases of a two-time-layer, 3-D CA are interpreted as physical properties. The CA simulates the laws of physics [4]. Wuensche's work, interpreting the time-series of CA updates and detecting attractors [11,12], also provides implicit interpretation layers. This work is potentially important for engineering emergence, since design is likely to be considerably facilitated if the attractors of an emergent feature can be established.

8 Conclusion

Our work exploits notions of layering and causality in emergent systems to improve our ability to engineer required properties and to enhance the expressive power of our simulations.

Having introduced layers for migrated rules, in the platelet model we can exploit the higher layer for more natural control laws, and the implemented platelet simulation could easily be extended to a two- or three-dimensional representation that is a better model of a blood vessel. We can introduce and experiment with models of environmental interaction, and, having abstracted platelet control from the CA grid, we could introduce local diffusion CAs on the absolute grid, modelling the chemical environment that acts on and is affected by the platelets. Further local CA rules could model flow features such as the effect of proximity to the vessel boundary on speed. These new features would be monitored by the higher-layer structures, which would also communicate "chemical" signals to the CAs.

In engineering terms, we are using layering and causality to devise architectural patterns for the design of emergent systems. We also seek to introduce good engineering practices, such as validation, testing and safety argumentation to the development process associated with this layered architecture.

Acknowledgements

H. R. Turner is funded jointly by EPSRC and BAE Systems. The work presented here is part of the TUNA feasibility study, EPSRC grant EP/C516966/1. Thanks are due to S. A. Schneider for the inspiration for the vacuum CA model.

References

1. P. B. Andersen, C. Emmeche, N. O. Finnemann, and P. V. Christiansen, editors. *Downward Causation. Minds, Bodies and Matter*. Åarhus University Press, 2000.
2. F.R.M. Barnes and P.H. Welch. Communicating Mobile Processes. In I. East, J. Martin, P. Welch, D. Duce, and M. Green, editors, *Communicating Process Architectures 2004*, volume 62 of *Concurrent Systems Engineering Series*, pages 201–218. IOS Press, September 2004.
3. F.R.M. Barnes, P.H. Welch, and A.T. Sampson. Barrier synchronisations for occam-pi. In *Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'2005)*, pages 20–26. CSREA press, 2005.
4. E. Fredkin. Digital mechanics: An informational process based on Reversible Universal CA. *Physica D*, 45:254–270, 1990.
5. R. Freitas. Clottocytes: artificial mechanical platelets. Technical Report IMM Report Number 18: Nanomedicine, Foresight Update 41, 2000.
6. M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223:120–123, 1970.
7. C.A.R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.
8. R. Milner. *The Pi Calculus*. Cambridge University Press, 1999.
9. F. Polack and S. Stepney. Emergent properties do not refine. In *REFINE 2005, BCS-FACS Refinement Workshop*, ENTCS. Springer, April 2005.
10. P.H. Welch and F.R.M. Barnes. Communicating mobile processes: introducing occam-pi. In A.E. Abdallah, C.B. Jones, and J.W. Sanders, editors, *25 Years of CSP*, volume 3525 of *LNCS*, pages 175–210. Springer, 2005.
11. A. Wuensche. Finding gliders in cellular automata. In A. Adamatzky, editor, *Collision-Based Computing*. Springer, 2002.
12. A. Wuensche. Discrete Dynamics Lab: Tools for investigating cellular automata and discrete dynamical networks. In A. Adamatzky and M. Komosinski, editors, *Artificial Life Models in Software*. Springer, 2004.

A Appendix: CA Model of Platelets

This is one possible CA rule set to simulate platelet clustering. The cell design has boolean `state`, `next` and `vacuum` variables. The `nd` variable is set to determine whether a cell containing a platelet loses its platelet in the next time step. It takes values 0, 1 and 2, where 0 represents "no change", 1 represents "change" and 2 is assigned if the resolution of non-determinism is a decision to change.

The value of `nd` is deterministic for all cells except the first cell of a cluster and the cell immediately behind (most) vacuums. The value is set using the first set of CA rules. The `next` variable of each cell is then calculated in the second phase. Calculations within a phase can be concurrent, as can the actual update where each `state` is reset to the cell's `next`.

A.1 The First Phase

The first pass visits only cells having `cell[i].state = TRUE`.

Rule A. A platelet cannot move because it is blocked by platelets ahead.
Rule B. A platelet cannot move because it is located ahead of a vacuum.
Rule C. A platelet between two vacuums cannot move.
Rule D. A platelet before a vacuum can decide whether or not to move.
Rule E. The platelet at the front of a cluster can decide whether or not to move.
Rule F. A singleton platelet must move.

The tabular summary gives the rule name, then the applicable values of the current cell and its neighbours, and the resulting `nd`. An occupied cell is labelled, T; an empty cell, -; and a vacuum, V.

RULE				nd
A	T	T	T	0
	-	T	T	0
	V	T	T	0

RULE				nd
B	V	T	-	0
C	V	T	V	0

RULE				nd
D	T	T	V	nd
	-	T	V	nd
E	T	T	-	nd

RULE				nd
F	-	T	-	1

A.2 The Second Phase

In the second pass, any cell that contains a platelet and has an `nd` value of 0 is unchanged. The `next` state for cells with `nd` > 0 is calculated to take account of vacuums. Cells that do not contain platelets have their state calculated.

Rule S.1. A cell with `nd` $= 2$, and a platelet behind, becomes a vacuum.
Rule S.2. A cell with `nd` $= 2$, which is at the back of a cluster, becomes empty.
Rule S.3. A cell holding a singleton platelet becomes empty.
Rule S.4. An empty cell, with an empty cell before it, does not change.
Rule S.5. An empty or vacuum cell, with a preceeding platelet having `nd` $= 0$, does not change. The CA design disallows a vacuum with an empty cell after it.
Rule S.6. A cell whose `nd` value is 0 does not change.
Rule S.7. An empty or vacuum cell with a platelet behind it having `nd` > 0, becomes occupied.

In the summary tables, each rule number is followed by the applicable states of the current cell and its neighbours; the last column is the `next` value of the current cell. Where the fact that the `nd` value was set non-deterministically is important, the resolved value is shown (eg T,2).

RULE				next
S.1.	T	T,2	-	V
	T	T,2	V	V
S.2.	-	T,2	V	-
S.3.	-	T	-	-

RULE				next
S.4.	-	-	-	-
	-	-	T	-
S.5.	T,0	-	T	-
	T,0	-	-	-
	T,0	V	T	V

RULE				next
S.6.	T	T,0	T	T
	-	T,0	T	T
	V	T	T	T
	V	T	-	T
S.7.	T,2	-	T	T
	T,2	-	-	T
	T,2	V	T	T

The Density Classification Problem for Multi-states Cellular Automata

Anna Rosa Gabriele

Università della Calabria, 87036 Arcavacata di Rende (CS), Italy
a.gabriele@unical.it
http://galileo.cincom.unical.it/esg/People/Anna_Rosa_Gabriele.html

Abstract. In this paper, the results of three experiments, in which a genetic algorithm evolves one-dimensional cellular automata (CA), in order to perform the classical main task, are reported. The used systems are not elementary CA but they have a higher number of states. Our aim is to verify if the main-task results are similar to those obtained with elementary CA. Our results confirm that there is a substantial homogeneity.

1 Introduction

CA have been used as models of biological systems as bugs' colonies, immune systems, brain's organization and economic systems, because they manifest emergent computation and complex behavior typical of these systems.

But how does this computation happen? What kind of behavior gives complex patterns of organization? One of the earlier problem in this topic has been the density classification task, introduced by Packard. Using CA with two states for analyzing their ability in performing computation, Packard [12] investigated the ability of a Cellular Automaton, starting from the density of the states in an arbitrary initial configuration, to compute the final configuration. So the CA rule is interpreted as a program, the initial configuration as an input, the final configuration obtained after a fixed number of steps as an output. This density classification problem consists in determining some rules that evolve the CA towards an homogeneous final configuration (composed uniquely of 1s or 0s) following the higher concentration of 1s or 0s in the initial configuration.

In the following years the group of researchers of the Santa Fe Institute [3], [4], [9], [10], [11] besides implementing a genetic algorithm to evolve rules able to solve the problem of the density classification, identified the emergency of computational strategies and analyzed the central role of symmetry in an evolutionary system. Particularly they shown as the break-up of symmetry can prevent the evolution to select rules with higher computational ability.

It has been shown that a rule for two-states one-dimensional CA, which correctly classifies all possible initial configurations [5], as in Packard, does not exist.

In 1996 Capcarrère, Sipper and Tommasini [1] demonstrated that a solution to the density classification problems does exist, defining a different output in

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 443–452, 2005.

comparison to that of Packard (*The output is not a fixed-point configuration but if the initial configuration's density is* > 0.5 *(respectively,* > 0.5*), the final configuration consists of one or more blocks of at least two consecutive 1s (0s), interspersed by an alternation of 0s and 1s; for an initial density of exactly 0.5, the final configuration consists of an alternation of 0s and 1s.*)

In 2001, Capcarrère and Sipper [2] demonstrated that a rule, which resolves the density classification problem, for a one-dimensional elementary CA, has to satisfy two conditions:

- the density of the initial configuration must be preserved over time.
- the rules table must exhibit density of 0.5".

In this work, in the first paragraph, a brief overview on Cellular Automata is given. Then, a genetic algorithm, implemented for analyzing the density classification task for multi-states CA, is described. Subsequently, some analysis and results obtained using the statistic parameter λ [6], [7] are shown, and at the end conclusions are drawn.

2 Cellular Automata and Genetic Algorithms: An Overview

The CA were introduced by von Neumann and Ulam as simple models for the study of some biological processes. A CA is a discreet dynamical system in which space, time and states assume discrete values. The space is represented by a n-dimensional regular grid ($n \in \mathbb{N}$), each element of the grid is called cell. Each cell can be considered as the basic element of CA and it contains a datum a_i^t that represents the state of the i^{th} cell at the time step t.

A cellular automaton is defined as a tuple:

$$CA = (d, S, N, \delta) \tag{1}$$

where d is a positive integer that indicates the CA dimension; S is a set of finite states ($|S| = k$); N is a vector $N = (x_1, x_2, \ldots, x_n)$ constituted by n elements that compose the neighborhood of each single cell; δ is a transition rule. This function characterizes the rule with which the CA evolves. The total number of rules depends on the number of states (k) and on the number of elements (n) which compose the neighborhood.

2.1 One-Dimensional Cellular Automata

In this article one-dimensional CA with three-states and with two-radius are considered. Formally, if a_i^t denotes the i^{th} cell value at the time step t of an one-dimensional CA, then

$$a_i^{t+1} = \delta(a_{i-r}^t, \ldots, a_i^t, \ldots, a_{i+r}^t) \tag{2}$$

The function δ will be completely defined when for every possible neighborhood ($C = k^{(2r+1)}$) a value β_i is assigned. The succession $(\beta_1, \beta_2, \ldots, \beta_C)$ so obtained

individualizes the *rule table* of the CA evolution or in general the *rule* of the CA. The same rule of a CA can be also represented expressly using a function based on a sum of the cells of the neighborhood. The state a_i^t is given from:

$$a_i^t = \delta\left(\sum_{j=-r}^{r} \alpha_j a_{i+j}^{t-1}\right) = \delta(\alpha_{-r}\ a_{i-r}^{t-1} + \ldots + \alpha_0\ a_i^{t-1} + \ldots + \alpha_r\ a_{i+r}^{t-1}) \qquad (3)$$

where α_j are constants integer.

Particularly, all rules, defined from (2), can be defined through (3) considering $\alpha_j = k^{r-j}$ (see [14]).

A CA configuration (c_t) is the set of the states in which all the cells are found at a particular time step t. Denoted with c_0 the initial configuration, the CA evolution can be represented bringing the sequence of configurations $\{c_t\}_{t>0}$ one following to the other. This sequence is called space-temporal diagram associated to the CA.

The experiments described in this article concern one-dimensional CA with $k = 3$ and $r = 2$, the neighborhood is composed from the same cell and from the two cells on the right and on the left.

3 Details of Cellular Automata and Genetic Algorithms in Our Experiments

In this article, the computational assignment for CA has been to identify, after a fixed number of steps M, if in the initial configuration more states in the condition 0 or more states in the condition 1 or 2 were present. Thus the output is an homogeneous configuration with only one state, the same one present in greater concentration in the initial configuration.

The CA, evolved through genetic algorithms, have $k = 3$ and $r = 2$ so the length of rule is equal to $3^{(2*2+1)} = 3^5 = 243$.

The initial conditions have a number of cells equal to $N = 151 = 50 \times 3 + 1$. N has been chosen not multiple of three, so to avoid situations of parity.

Three experiments have been realized, composed by 30 runs with 100 generations each, that differ among them for the fitness function or for the procedure by which rules of every population have been produced.

Following a standard method, a genetic algorithm [8], with an initial population of 231 rules randomly produced, has been implemented for each experiment. These rules have been tested on 231 randomly produced initial conditions at every evolution.

The rules and the initial conditions have been produced choosing 77 of each of them in which the maximum number of elements is 0, 77 in which the maximum number of elements is 1 and 77 in which the maximum number of elements is 2. For each group of 77 rules or initial conditions, the probability of the presence of cells or local rules in the maximum condition is uniformly distributed according to Mitchell and colleagues [9].

Each rule has been tested on 231 initial conditions for a temporal evolution of M steps, M is calculated using the following:

$$M = 453 + \lfloor r \rfloor \qquad r \in [0, 3] \tag{4}$$

where r is chosen randomly in $[0, 3]$ and $\lfloor \cdot \rfloor$ denotes the integer part of a number.

For every generation the algorithm performs the followings steps:

- a new set of initial conditions is generated
- fitness is evaluated for each rule
- mean fitness is evaluated
- 20 % of the best rules (élite) is copied and the remainder 80% is modified in the following ways:
  - the élite is randomly crossed (i.e. crossovers between randomly chosen pairs of elite rules)
  - each site is changed with a 3% of probability.
- the preceding rules are replaced with the new ones.

At the end of the experiments the performance of the best rule has been verified on 231 new initial conditions randomly produced following the criteria previously described.

Some statistical analysis on the best rule have been done according to Langton [6], [7]. The statistical parameter λ_i has been defined considering the quiescent state respectively equal to 0, 1 and 2, and it has been calculated using the following expression:

$$\lambda_i = \frac{\text{number of elements different from i in the rule table}}{\text{total number of elements in the rule table}} \tag{5}$$

with $i = 0, 1, 2$.

3.1 Experimental Set 1

According to Mitchell and colleagues [9], in the first experiment, the following fitness function has been used:

$$f_i = \sum_{j=1}^{231} \frac{f_{ij}}{231}$$

where f_i is the fitness value related to the i^{th} rule and f_{ij} is calculated for each initial condition j in the following way:

$$f_{ij} = \begin{cases} 1 & \text{if the final pattern is correct} \\ 0 & \text{in the other cases.} \end{cases}$$

During the first experiment the best individual resulted in the 4th run, 98th generation, with a percentage of success of 89%. The evolution fitness is represented in figure 1a. In figure 1b three histograms in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2 are shown. In the figure 1c the best evolved rule is shown.

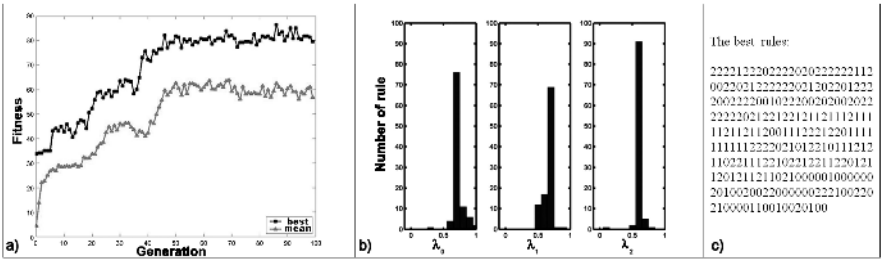


Fig. 1. In (a): Fitness evolution of the best experiment. In (b): three histograms are shown in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2. In (c): the best rule has the following values: $\lambda_0 = 0.7$, $\lambda_1 = 0.7$ and $\lambda_2 = 0.6$.

3.2 Experimental Set 2

In the second experiment a genetic algorithm has been implemented with an initial population of 231 rules, randomly produced, in which the local rules, that correspond to the neighborhoods 22222, 11111, 00000, are fixed respectively to the values 2, 1 and 0 (see Lemma 1.2 [2]).

The same function of the first experiment has been used as fitness function.

The best individual of the second experiment resulted at the 12th run (in Figure 2a the fitness evolution), at the 76th generation, with a percentage of success of 87%. In figure 2b three histograms in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2 are shown. In figure 2c the best rule is shown.

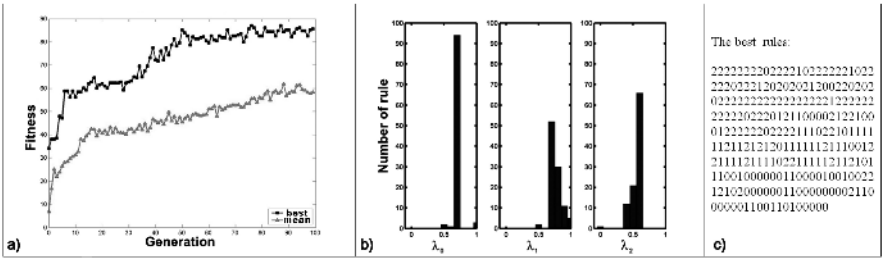


Fig. 2. In (a): Fitness evolution of the best experiment. In (b): three histograms are shown in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2. In (c): the best rule has the following values: $\lambda_0 = 0.7$, $\lambda_1 = 0.7$ and $\lambda_2 = 0.6$.

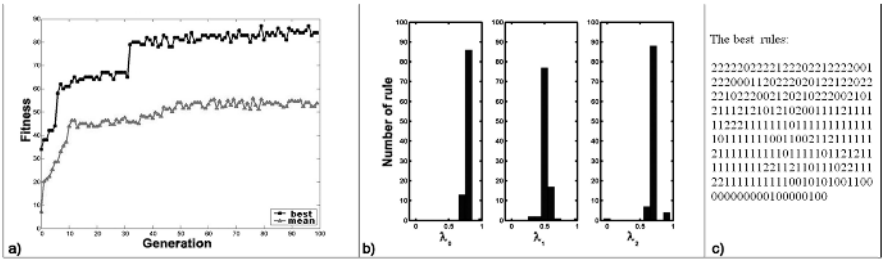


Fig. 3. In (a): Fitness evolution of the best experiment; In (b): three histograms are exposed in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2. In (c): the best rule has the following values: $\lambda_0 = 0.8$, $\lambda_1 = 0.5$ and $\lambda_2 = 0.7$.

3.3 Experimental Set 3

The fitness function of the third experiment is:

$$f_i = \sum_{j=1}^{231} \frac{f_{ij}}{231}$$

where f_i is the fitness value related to the i^{th} rule and f_{ij} is calculated for each initial condition j in the following way:

$$f_{ij} = \frac{\text{number of cells in the correct states}}{\text{total number of cells}}$$

During this experiment the best rule resulted at the 4th run, 79th generation, with a percentage of success of 87%.

The fitness evolution is shown in figure 3a. In figure 3b three histograms in which the distribution of the number of rules is represented as a function of the parameters λ_0, λ_1 and λ_2 are shown. In figure 3c the best rule is shown.

4 Performance of Best Rules

Observing the performance of the best rules evolved with the three experiments, the percentage of success is still very elevated in all experiments. The best rules of the first and the second experiments classify very well the initial conditions in which maximum density is of 0s and 1s, while they classify with more difficulties initial conditions with maximum density of 2s (Figure 4(a) and (b)). The best rule of the third experiment classifies very well the initial conditions in which maximum density is 2s and 0s, while classifies with more difficulties initial conditions with maximum density of 1s (Figure 4(c)).

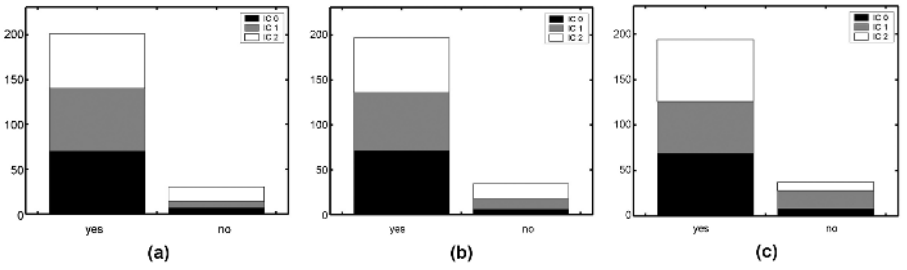


Fig. 4. In these histograms the number of rules, that classify (yes) or not (no) initial conditions, are shown. IC 0, IC 1 and IC 2 respectively point out initial conditions with maximum densities of 0s, 1s and 2s. The percentage of success of the best rule of the first experiment is around 88%, of the best rule of the second experiment is around 85% and of the best rule of the third experiment is around 84%.

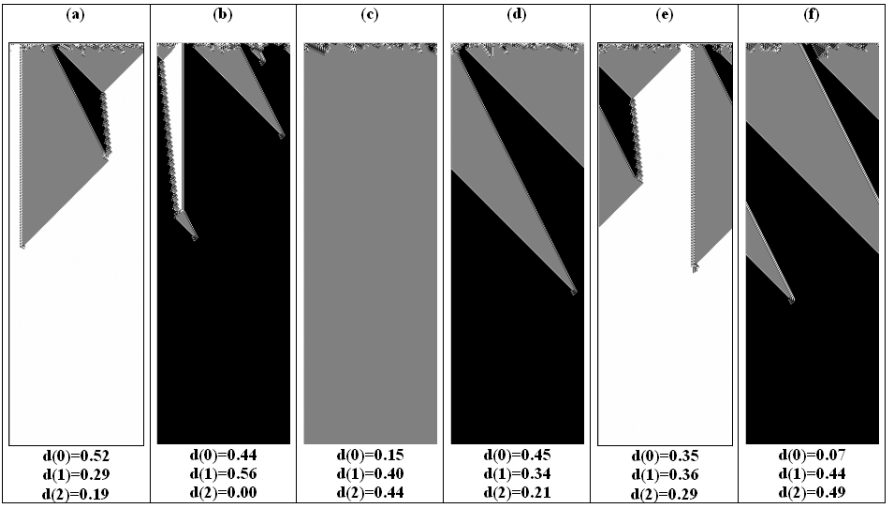


Fig. 5. Diagrams obtained evolving CA with the best rule of the first experiment

In figures 5, 6 and 7 space-temporal diagrams of CA with k=3 r=2 are shown. These diagrams are obtained for different initial configurations over 453 time steps. The quantities $d(0)$, $d(1)$, $d(2)$ indicate the density of 0s, 1s and 2s in initial conditions. In (a), (b) and (c) the CA classified the initial configurations according to their density, while, in (d), (e) and (f) the rule don't classified any configuration.

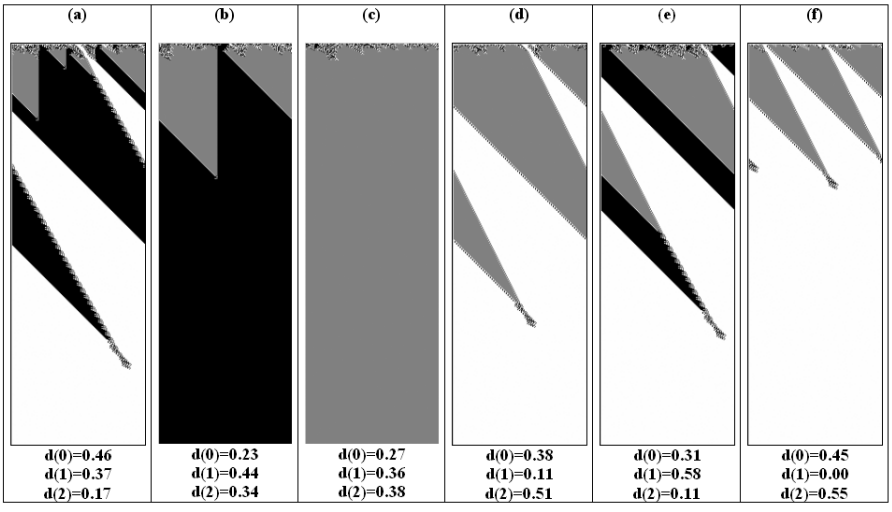


Fig. 6. Diagrams obtained evolving CA with the best rule of the second experiment

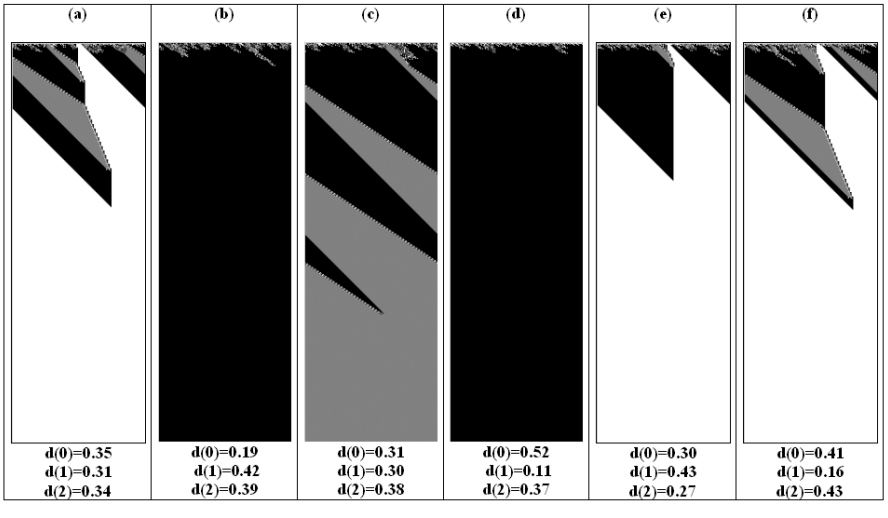


Fig. 7. Diagrams obtained evolving CA with the best rule of the third experiment

5 Conclusions

A genetic algorithm, that selects rules that classify initial conditions with a higher probability, has been developed. In all experiment the obtained percentage of success is between 87% and 89%. These results are also confirmed in the test phase, in which the obtained percentage of success is between 84% and 88%. Observing the distribution of the values λ_0, λ_1, and λ_2 in the two first

experiments, the values of the local rules are uniformly distributed, while in the third experiment a great presence of local rules in a particular state is observed.

Observing the space-temporal diagrams obtained by the three experiments, it can be noticed that, in the first two experiments, the best evolved rules classify very well initial conditions with maximum density of 0s and 1s. The strategy that they use is to expand the blocks of 1s and 0s present in the initial conditions. About the initial conditions in which maximum density is of 2s, it can be observed that the CA classify them in few time steps. The time steps, obtained in the first experiment, are of maximum 19 against the 259 steps necessary to classify initial conditions with maximum density of 0s and against the 302 steps necessary to classify initial conditions with maximum density of 1s. About the second experiment the time steps necessary to classify the initial conditions in which maximum density is of 2s are 25 against the 368 for the maximum density of 0s and against the 155 for the maximum density of 1s. About the third experiment, the results are different from the previous ones: the evolved rule, in fact, classifies very well initial conditions with maximum density of 0s and of 2s, while, for the initial conditions in which maximum density is of 1s, the same rule classifies them in few time steps (maximum time 28 steps). Observing the space-temporal diagrams obtained in the third experiment it can be noticed that the best evolved rule uses the following strategy: it expands the blocks of 0s and 2s of the initial conditions, with different speed (see figure 7a and 7c).

The results previously exposed have been obtained from the first analysis. Some strategies, that have been individualized in the evolution of CA with $k = 3$ and $r = 2$, are very similar to those obtained in the evolutions of CA with $k = 2$.

Further analysis, to individualize and to formalize the mechanisms of computation in the evolution of CA with k=3 r=2, is currently in progress.

References

1. Capcarrére M. S., Sipper M., and Tommasini M.: Two-state, r=1 Cellular Automaton that Classifies Density. Physical Review Letters **77** (1996) 4969-4971.
2. Capcarrére M. S. and Sipper M.: Necessary conditions for density classification by cellular automata. Physical Review E **64** (2001) 036113/1–036113/4.
3. Crutchfield J. P. and Mitchell M.: The Evolution of Emergent Computation. Proceedings of the National Academy of Sciences **92** 23 (1995) 10742–10746.
4. Hordijk W., Crutchfield J. P. and Mitchell M.: Mechanisms of Emergent Computation in Cellular Automata. In Parallel Problem Solving from Nature-V, A. E. Eiben, T. Bck, M. Schoenauer, and H.-P. Schwefel (eds.) Springer-Verlag (1998) 613-622.
5. Land M. W. S. and Belew R. K.: No Two-State Ca for density classification exists. Physical Review Letters **74** 25 (1995) 5148–5150.
6. Langton C. G.: Studying Artificial Life With Cellular Automata. Physica D **22** (1986) 120–149.
7. Langton C. G.: Computation at the edge of Chaos, Phase Transitions and Emergent Computation. Physica D **42**(1990) 12–37.
8. Mitchell M.: An Introduction to Genetic Algorithms. MIT Press (1996).

9. Mitchell M., Hraber P. T., and Crutchfield J. P.: Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations. Complex Systems **7** (1993) 89–130.
10. Mitchell M., Crutchfield J. P. and Hraber P. T.: Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments. Physica D **75** (1994) 361–391.
11. Mitchell M., Crutchfield J. P. and Das R.: Evolving cellular automata to perform computations: A review of recent work. In in Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA '96), Russian Academy of Sciences (1996) 42-55.
12. Packard N. H.: Adaptation toward the edge of chaos. In Dynamic Patterns in Complex Systems, J. A. S. Kelso, A. J. Mandell, and M. F. Shlesinger (eds) World Scientific (1988) 293–301.
13. Sipper M., Capcarrère M. S., and Ronald. E.: A simple cellular automata that solves the density and ordering problems. International Journal of Modern Physics C **9** 7 (1998) 899-902.
14. Wolfram S.: Universality and Complexity in Cellular Automata. Physica D **10** (1984) 1-35.

Evolving Cellular Automata by $1/f$ Noise

Shigeru Ninagawa

Division of Information and Computer Science,
Kanazawa Institute of Technology,
Ohgigaoka, Nonoichi, Ishikawa 921-8501, Japan
ninagawa@infor.kanazawa-it.ac.jp

Abstract. It is speculated that there is a relationship between $1/f$ noise and computational universality in two-dimensional cellular automata. We use genetic algorithms to find two-dimensional cellular automata which have $1/f$ spectrum. Spectrum is calculated from the evolution of the state of cell from a random initial configuration. The fitness function is constructed in consideration of the spectral similarity to $1/f$ spectrum. The result shows that the rule with the third highest fitness in the experiment has $1/f$ spectrum and it behaves like the Game of Life, although two rules with the highest and the second highest fitness do not have $1/f$ spectrum.

1 Introduction

The Game of Life (LIFE) [1] is one of the two-dimensional cellular automata (CAs). Although the rule for the evolution in LIFE is very simple, it generates complicated patterns such as a glider which propagates infinitely until it is annihilated when it collides with another object on the array. It is supposed that a universal computer can be constructed on the array by considering a glider as a pulse in a digital circuit.

Moreover, LIFE is characterized by $1/f$ noise [2]. The spectra calculated from the evolution of cells from a random initial configuration exhibit $1/f$ spectrum in LIFE. $1/f$ noise is a random process whose spectrum S_f as a function of the frequency f behaves like $1/f^{\beta}$ with $\beta \approx 1$ at low frequencies. $1/f$ noise has been observed in many different systems, but its origin is not well understood [3].

These results suggest that there is a relationship between computational universality and $1/f$ noise in cellular automata. Since CAs which exhibit $1/f$ spectrum have not been found except for LIFE, we need to find those kind of CAs to verify the relationship.

Genetic algorithms (GAs) have been used to discover CAs with desirable properties [4,5]. In their work they searched for one-dimensional CAs which can classify the density of the initial configurations and then could find CAs with high performance on the task. In this paper we apply GAs to searching for two-dimensional CAs which have $1/f$ spectrum.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 453–460, 2005.

2 $1/f$ Noise in LIFE

A two-dimensional CA is a lattice system which evolves in discrete time steps. Every site takes on state 0 or state 1 at any one time step and is updated synchronously according to a rule. Let $s_{x,y}(t)$ denote the state of the cell at position (x, y) at time step t. The state of the site (x, y) evolves by the rule function d,

$$s_{x,y}(t+1) = d(s_{x,y}(t), n_{x,y}(t)), \tag{1}$$

where $n_{x,y}(t)$ denotes the sum of the states of the eight nearest neighboring sites around the site (x, y) at time step t. The rule of LIFE is defined by

$$\begin{aligned} d(0,3) = d(1,2) = d(1,3) = 1, \\ otherwise \quad d = 0. \end{aligned} \tag{2}$$

Spectral analysis is one of the useful methods to investigate the behavior of dynamical systems [6]. While it was used for the analysis of the spatial structure produced by one-dimensional CAs [7], we apply it to the analysis of the temporal behavior of two-dimensional CAs.

The Fourier transformation of a evolution of states $s_{x,y}(t)$ of the site (x, y) for $t = 0, 1, ..., T-1$ is given by

$$\begin{aligned} \hat{s}_{x,y}(f) = \frac{1}{T} \sum_{t=0}^{T-1} s_{x,y}(t) \exp(-i \frac{2\pi t f}{T}) \\ (f = 0, 1, \cdots, T-1). \end{aligned} \tag{3}$$

The spectrum is defined as

$$S_f = \sum_{x,y} |\hat{s}_{x,y}(f)|^2, \tag{4}$$

where the summation is taken over all cells in the array. The power S_f at frequency f intuitively means the "strength" of the periodic vibration with period T/f in the evolution in T time steps.

The least square fitting

$$\ln(S_f) = \alpha + \beta \ln(f), \tag{5}$$

of the observed spectrum S_f from $f = 1$ to $f = f_b$ gives the coefficient α and β. The residual sum of squares σ^2 is given by

$$\sigma^2 = \frac{1}{N_r} \sum_{f=1}^{N_r} (\ln(S_f) - \alpha - \beta \ln(f))^2, \tag{6}$$

where N_r is the number of data used for the calculation of σ^2.

Throughout this paper the array consists of 100×100 sites and periodic boundary conditions are used. The array is started from a random initial configuration in which each site takes state 0 or state 1 randomly with independent

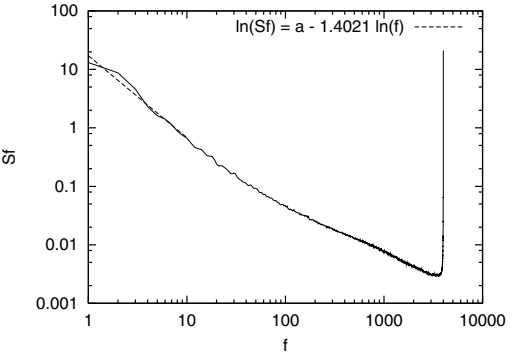


Fig. 1. Spectrum of LIFE for T = 8000 from a 100×100 random initial configuration. The dotted line represents the least square fitting of the spectrum by $\ln(S_f) = \alpha + \beta \ln(f)$ from $f = 1$ to $f = 10$ with $\beta = -1.4021$.

equal probabilities. The density of cell of state 1 of the initial configuration is actually 0.4978.

The spectrum of LIFE calculated by (4) for $T = 8000$ is shown in Fig.1. The dotted line in Fig.1 represents the least square fitting of the spectrum according to (5) for $f_b = 10$ with $\alpha = 2.8412 \pm 0.1283$ and $\beta = -1.4021 \pm 0.0772$. This spectrum behaves like $1/f^{\beta}$ with $\beta \approx 1$ at low frequencies and it is considered to be $1/f$ noise.

3 Experiment

3.1 Fitness

In this paper we use GAs to evolve two-dimensional two-state nine-neighbor outer totalistic CAs whose spectra exhibit $1/f$ noise. We encode a rule in (1) as follows:

$$d(0, i) = x_{2i},\ d(1, i) = x_{2i+1}\ (i = 0, 1, \cdots, 8). \tag{7}$$

Since we consider the state 0 as the quiescent state, we restrict the rules to those with $d(0, 0) = 0$. Therefore a rule is represented by a 17 bit string $x_{17}x_{16}\cdots x_1$. The rule of LIFE (2) is expressed by "00000000001110000" in this representation.

The fitness of a rule is calculated from the shape of its spectrum. The fitness is given by: (i) calculating the spectrum S_f of the rule by (4); (ii) calculating the coefficient β and the residual sum of squares σ^2 by (5), (6); (iii) calculating the fitness F by

$$F = \frac{|\beta|}{\sigma^2 + \delta}, \tag{8}$$

where δ is the correction term to avoid division by zero and is set to 1×10^{-6} throughout this research.

The numerator is intended that the smaller the coefficient β at low frequencies is, the larger the numerator is. The rule with positive coefficient β is rarely generated, and moreover, the positive coefficient β is almost always small. The denominator is supposed to guarantee that the spectrum fits the power law (5) in a broad range of frequencies. Therefore we set $f_b = 10$, $N_r = 3000$ in this experiment. By using this fitness function, we hope that the closer to $1/f$ spectrum the spectrum is, the higher fitness the rule has. The residual sum of squares σ^2 of LIFE calculated according to (6) from the spectrum shown in Fig. 1 is 0.00726 and the fitness F is 193.14.

3.2 Details of the Experiment

CA rules are parameterized by a parameter λ which is the fraction of nonzero output states in the rule table [8]. Generally speaking as $\lambda = 0$ varies from 0 to $1 - (1/K)$ (K is the number of cell states), CAs change from the most homogeneous rule to the most heterogeneous rule. So we randomly generate the rules whose λ is uniformly distributed between 1/18 and 9/18 in an initial population.

The evolution from a random initial configuration in the square array 100×100 in periodic boundary conditions leads to periodic configurations in about 2000 time steps on average through transient behavior in LIFE [9]. $1/f$ spectrum in CAs is caused by the transient behavior from random initial configurations. Therefore, the longer the duration of observation T in (3) becomes, the more the spectrum deviates from $1/f$ spectrum especially at low frequencies. When T becomes over 7200 in the square array 100×100 in periodic boundary conditions in LIFE, the spectra become level at low frequency [10]. The spectrum in Fig. 1 is a typical example where power density becomes somewhat close to a level at the frequency $f = 2$ or below because it is calculated for $T = 8000$.

In this research we set T at 8000 to find the CAs comparable or more in transient length to LIFE. But the calculation of spectrum for $T = 8000$ needs a lot of time. So we carry out a preliminary selection from randomly generated rules to remove the rules whose spectrum is far from $1/f$ spectrum. In the preliminary selection the spectra S_f for $T = 1024$ of randomly generated rules are calculated. We pick the rules with $\beta \leq -0.3$ in (5) for $f_b = 400$.

Our experiment proceeds as follows.

1. A population of rules with λ varying between 1/18 and 9/18 is randomly generated.
2. The rules with $\beta \leq -0.3$ by the least square fitting of the spectrum for $T = 1024$ and $f_b = 400$ are selected and are gathered in a population of P rules.
3. F is calculated with $T = 8000$ for each rule in the population.
4. A number E of the highest fitness rules is copied without modification to the next generation.
5. The remaining $P - E$ rules for the next generation are formed by uniform crossovers with a probability of P_c between pairs in the population chosen

by roulette wheel selection. Every bit of the offspring from each crossover are mutated with a probability of P_m.

One generation consists of steps 3 - 5 and it is repeated several times for one run. Our experiment is composed of 30 runs with same parameters except for random number seed. The number of the generations repeated in each run is not identical because our experiment is in progress. We set $P = 140$, $E = 10$, $P_c = 0.6$, and $P_m = 0.03$.

Since the fitness of rule is depend on initial configurations, it is reasonable to vary initial configuration in each generation. However, in our experiment the values of b and σ^2 of every generated rule are recorded in a file to use in calculating the fitness of the same rule in later generations. By using this approach instead of computing all fitness in every generation, considerable computation time is saved. Therefore we use only one initial configuration through the evolution in GA.

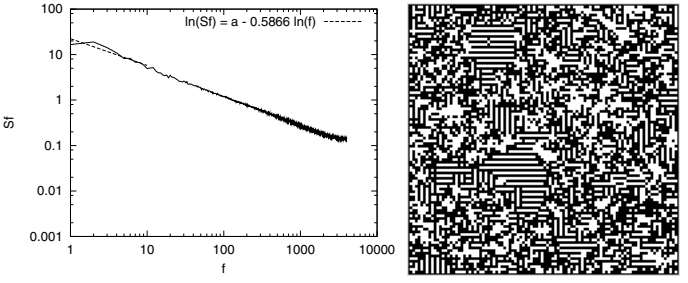
3.3 Results

We have performed the experiments for a total of 1800 generations in 30 runs. Figure 2 (a) shows the spectrum of the rule "01000011101110000" with the highest fitness $F = 996.09$ ($\beta = -0.5866$, $\sigma^2 = 0.00059$) in the experiment. The dotted line in the spectrum represents the least square fitting of the spectrum according to (5) from $f = 1$ to $f = 10$. Figure 2 (b) shows the pattern at time step $t = 1000$ generated from the same initial configuration as in Fig. 2 (a). White squares represent cells with state 0 and black squares represent cells with state 1. We call this rule F1. The observation of the evolutions from random initial configurations shows that the state of cells changes abruptly and the fixed patterns like a maze are gradually formed. The average of b, σ^2, and F of F1 for ten distinct initial configurations is -0.7857, 0.0010, and 951.04.

Figure 3 (a) shows the spectrum of the rule "10110110000110100" with the second highest fitness $F = 661.18$ ($\beta = -0.2819$, $\sigma^2 = 0.00043$). Figure 3 (b) shows the pattern at time step $t = 1000$ generated from the same initial configuration. We call this rule F2. The state of cells changes abruptly and the clusters of cells with state 1 are gradually formed in the evolution. The average of b, σ^2, and F of F2 for the ten distinct initial configurations is -0.2138, 0.0005, and 433.53.

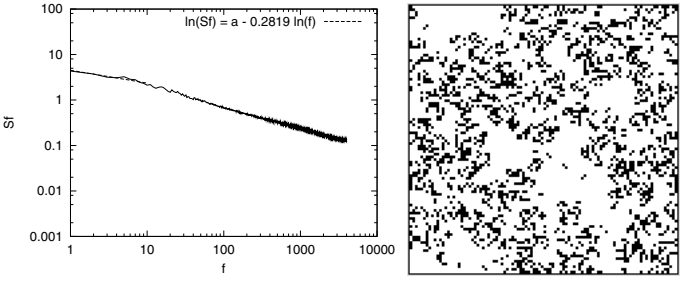
Although F1 and F2 have high fitness, the exponent β in these spectra is not close to -1. Therefore these are not considered to be $1/f$ noise. The highness of fitness is primarily due to the lowness of the residual sum of squares σ^2 in these spectra. There seems to be no propagating structures in F1, although there are 7 gliders in F2 [11].

Figure 4 (a) shows the spectrum of the rule "01010000001110000" with the third highest fitness $F = 573.85$ ($\beta = -1.0508$, $\sigma^2 = 0.0018$). Figure 4 (b) shows the pattern at time step $t = 1000$ generated from the same initial configuration. We call this rule F3. The evolution of F3 from random initial configurations



(a) (b)

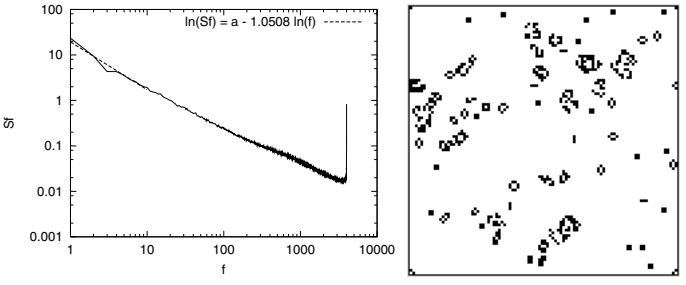
Fig. 2. (a) Spectrum for T = 8000 from a 100×100 random initial configuration of the rule "01000011101110000" with the highest fitness $F = 996.09$. The dotted line represents the least square fitting of the spectrum by $\ln(S_f) = \alpha + \beta \ln(f)$ from $f = 1$ to $f = 10$ with $\beta = -0.5866$. (b) Pattern at time step $t = 1000$ generated by the rule from the same initial configuration. White squares represent cells with state 0 and black squares represent cells with state 1.



(a) (b)

Fig. 3. (a) Spectrum for T = 8000 from a 100×100 random initial configuration of the rule "10110110000110100" with the second highest fitness $F = 661.18$. The dotted line represents the least square fitting of the spectrum by $\ln(S_f) = \alpha + \beta \ln(f)$ from $f = 1$ to $f = 10$ with $\beta = -0.2819$. (b) Pattern at time step $t = 1000$ generated by the rule from the same initial configuration.

is fairly similar to that of LIFE, and moreover, there is the same glider in F3 as in LIFE. The rule of F3 is different from that of LIFE in two output states $x_{16} = x_{14} = 1$ which correspond to $d(0, 8) = d(0, 7) = 1$. This means cells in state 0 in F3 tend to turn to state 1 as compared with LIFE and the cluster of cells in state 1 lasts a long time, changing its shape. Therefore the transient behavior from a random initial configuration in F3 lasts longer than in LIFE. The average of b, σ^2, and F of F3 for the ten distinct initial configurations is -1.0417, 0.0024, and 440.10.



(a) (b)

Fig. 4. (a) Spectrum for T = 8000 from a 100×100 random initial configuration of the rule "01010000001110000" with the third highest fitness $F = 573.85$. The dotted line in the spectrum represents the least square fitting of the spectrum by $\ln(S_f) = \alpha + \beta \ln(f)$ from $f = 1$ to $f = 10$ with $\beta = -1.0508$. (b) Pattern at time step $t = 1000$ generated by the rule from the same initial configuration.

4 Conclusion

In this paper we reported the experiment in progress using GAs to find two-dimensional two-state nine-neighbor outer totalistic CAs with $1/f$ spectrum. F3 with the third highest fitness in the experiment has $1/f$ spectrum. While the rule of F3 is different in two output states $d(0, 8)$ and $d(0, 7)$ from that of LIFE, its behavior is extremely similar to that of LIFE, and moreover, there is the same glider as in LIFE. If F3 is capable of universal computation like LIFE, it can be an evidence supporting the relationship between computational universality and $1/f$ noise in two-dimensional CAs, although the detailed investigation will be needed to prove the ability for universal computation in F3, and moreover, we have to perform GA operations for longer generations. The rules with the highest and the second highest fitness in the experiment do not have $1/f$ spectrum. This defect suggests that there is still room for improvement in the fitness function (8).

The hypothesis of "the edge of chaos" has evoked considerable controversy [8]. This hypothesis says the ability to perform universal computation in a system arises near a transition from regular behavior to chaotic behavior. It is still uncertain whether the rules with 1/f spectrum are located near the transition to chaos.

In this research we concentrated on two-dimensional CAs. The elementary (one-dimensional two-state three-neighbor) CA rule 110 is capable of universal computation [12]. None of the elementary CAs show 1/f spectrum, although the spectra of the rule 110 and rule 54 have a remarkable feature that they have both power density in a broad range of frequencies like chaotic rules and several peaks in some frequencies like periodic rules [13]. Another criterion might be needed for computational universality in one-dimensional CAs instead of $1/f$ noise.

The proposed method can be applied to the search through larger CAs rule space than the CAs rule space dealt with in this article. The search through the two-dimensional three-state nine-neighbor outer totalistic CAs rule space will be performed in future work.

Acknowledgements. This study was carried out under the ISM Cooperative Research Program (2005-ISM·CRP-0007).

References

1. Berlekamp, E.R., Conway, J.H., Guy, R.K.: Winning Ways for Your Mathematical Plays, Vol.2, Academic Press, New York (1982)
2. Ninagawa, S., Yoneda, M., Hirose, S.: $1/f$ Fluctuation in the "Game of Life". Physica D **118** (1988) 49–52
3. Keshner, M.S.: $1/f$ Noise. Proc. IEEE **70** (1982) 211–218
4. Mitchell, M., Hraber, P.T., Crutchfield, J.P.: Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations. Complex Systems **7** (1993) 89–130
5. Mitchell, M., Crutchfield, J.P., Hraber, P.T.: Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments. Physica D **75** (1994) 361–391
6. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C 2nd ed., chapter 13, Cambridge University Press, Cambridge (1992)
7. Li, W.: Power Spectra of Regular Languages and Cellular Automata. Complex Systems **1** (1987) 107–130
8. Langton, C.: Computation at the Edge of Chaos: Phase Transitions and Emergent Computation. Physica D **42** (1990) 12–37
9. Ninagawa, S.: Cascade Process in the Transient Behavior of the "Game of Life". Proceedings of the Seventh International Symposium on Artificial Life and Robotics **16** (2002) 124–127
10. Ninagawa, S.: $1/f$ Fluctuation and Transient Behavior in the Game of Life. IPSJ Journal **43** (2002) 2017–2020 (in Japanese)
11. http://www.ics.uci.edu/~eppstein/ca/
12. Cook, M.: Universality in Elementary Cellular Automata. Complex Systems **15** (2004) 1–40
13. Ninagawa, S., Hirose, S., Hase, H., Yoneda, M.: Classification of One-dimensional Cellular Automata by Spectral Analysis. Trans. of the IEICE D-1 **J80** (1997) 856–865 (in Japanese)

Evolving Sequential Combinations of Elementary Cellular Automata Rules

Claudio L.M. Martins and Pedro P.B. de Oliveira

Universidade Presbiteriana Mackenzie
Rua da Consolação 896, Consolação
01302-907 São Paulo, SP – Brazil
claudio.luis.martins@terra.com.br
pedrob@mackenzie.br

Abstract. Performing computations with cellular automata, individually or arranged in space or time, opens up new conceptual issues in emergent, artificial life type forms of computation, and opens up the possibility of novel technological advances. Here, a methodology for combining sequences of elementary cellular automata is presented, in order to perform a given computation. The problem at study is the well-known density classification task that consists of determining the most frequent bit in a binary string. The methodology relies on an evolutionary algorithm, together with analyses driven by background knowledge on dynamical behaviour of the rules and their parametric estimates, as well as those associated with the formal behaviour characterisation of the rules involved. The resulting methodology builds upon a previous approach available in the literature, and shows its efficacy by leading to 2 rule combinations already known, and to additional 26, apparently unknown so far.

1 Introduction: Background and Motivation

1.1 Cellular Automata

Cellular automata (CAs) are fully discrete, complex systems that possess both a dynamic and a computational nature. They consist of a grid-like regular lattice of cells, and a state transition rule [2]. The cells in the lattice have an identical pattern of local connections to other cells, and are subjected to some boundary condition, usually periodic. Each cell can take on one of a discrete set of possible states, and the neighbourhood of a cell is defined as the cell, together with the others that are connected to it.

The state transition rule yields the next state for each cell, as a function of its neighbourhood, and, at each time step, all cells synchronously have their states updated. In computational terms, a cellular automaton is, therefore, an array of finite automata, where the state of each automaton depends on the state of its neighbours.

For one-dimensional CAs, the size m of the neighbourhood is usually written as $m=2r+1$, where r is called the radius of the automaton. In the case of binary-state CAs, the transition rule is given by a state transition table, which lists each possible neighbourhood together with its output bit, that is, the updated value for the state of the central cell in the neighbourhood. Figure 1 gives an example, with rule 110 of the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 461 – 470, 2005.

elementary space – the set of one-dimensional cellular automata rules with 2 states per cell and radius 1 – in which black cells represent state 1, and white cells represent 0. The rule denomination as 110 comes from the decimal number corresponding to the binary number that is formed from its rule table, from neighbourhood 111…1 on the left-hand side, as shown in the figure; in fact, such a naming scheme is widely used in the literature, for any radius, so that here we preserve it.

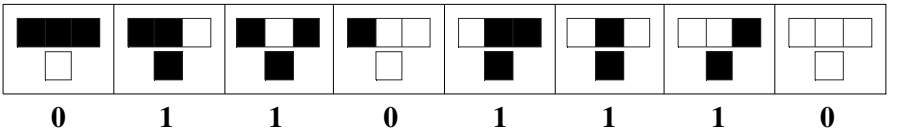


Fig. 1. Example of a cellular automaton rule

The elementary space is the most well-studied space in the literature, being composed of only 256 rules. Although small, it is very important, because it is extremely rich in its phenomenology and conceptual connections and implications; for instance it has recently been shown that rule 110 has universal computability [2], a quite surprising and remarkable result, considering such a rule is an extremely simple computational system.

The dynamics of a cellular automaton is associated with its transition rule. Figure 2 illustrates the possible regimes (the initial condition being at the top, and time running downward).

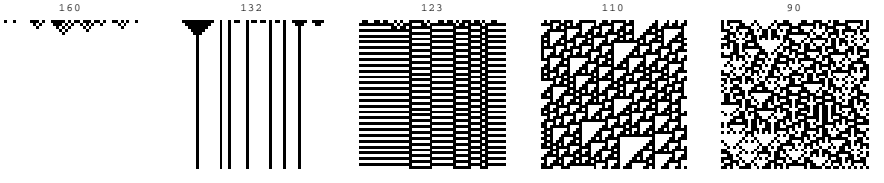


Fig. 2. Examples of the type of dynamical regimes in cellular automata: null, fixed point, periodic, complex and chaotic [13]

Taken from left to right, the pictures refer to the temporal evolution of elementary rules 160, 132, 123, 110 and 90, from random initial conditions, representing, respectively, the regimes: null (the limit configuration becomes homogeneous, all 1s or 0s in the binary case), fixed point, periodic, complex (the limit configuration becomes a mix of ordered and disordered regions) and chaotic.

Because it has been proven that the decision problem associated with predicting the dynamical behaviour of an arbitrary CA, with arbitrary initial condition, is an undecidable problem, computationally cheap ways of forecasting the dynamical behaviour of CAs have been conceived. Along this line, static parameters whose values can be directly derived from the transition rule of a CA have been defined, such as *sensitivity*, *absolute activity*, *neighbourhood dominance* and *activity propagation* [11].

Finally, still in respect to the dynamical behaviour of CAs, it is important to notice that the rules of a certain family – for instance, the 256 rules of the elementary space

– can be partitioned into *classes of equivalent dynamical behaviour*. This is achieved by changing all 0s to 1s in the rule table of a rule (the black-white transformation), by reversing all neighbourhoods while preserving the original output bit they originally lead to (the left-right transformation), and by doing the latter two in sequence. As a consequence, equivalence classes are formed with 4, 2 or a single rule. Figure 3 illustrates the dynamical equivalence class for elementary rule 110, formed by rules 137, 124 and 193, which are obtained, respectively, from each of the previous transformations.

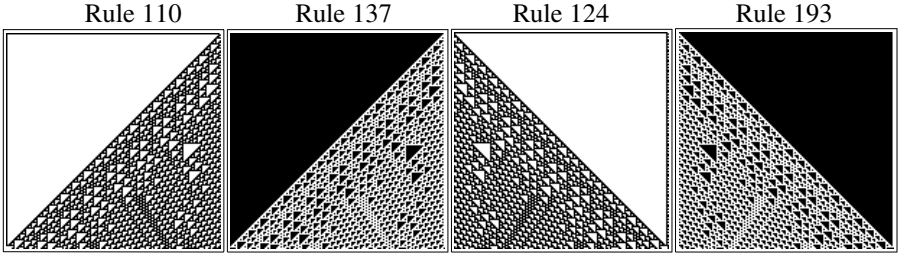


Fig. 3. Dynamical equivalence among CA rules is defined by the following symmetries in their rule tables: black-white, left-right, and the combination of both

1.2 Density Classification with Cellular Automata

A strong motivation for studying cellular automata is their ability to perform computations, through their characteristic totally decentralised, local and parallel mode [16]. However, the understanding of how these computations are carried out is still extremely vague, so that, regardless of more than four decades of cellular automata research, their use for computing functions at large are still at an embryonic stage. The main reason for this failure is the lack of a robust method for designing cellular automata of a predefined behaviour.

The *Density Classification Task* (DCT, for short) is one of the most widely studied computational problems in the context of cellular automata. In its standard formulation it states that a binary, one-dimensional CA has to converge to a final configuration of all cells in state 1, when the initial configuration has more 1s than 0s, and to a configuration of all 0s, whenever the initial configuration has more 0s than 1s. The problem usually does not specify what should happen to the CA if the initial configuration has as many ones as zeros, although one could require (as sometimes happens in the literature) that, in this case, a specific final configuration also has to be achieved (for instance, a binary sequence of a single 0 alternating with a single 1). While solving the DCT is a trivial task for any centralised computational system, it is a daunting task for any fully distributed system, with local processing, as a cellular automaton, in that it requires global coordination to be solved, thus being a clear example of emergent computation.

Although the DCT was proposed in 1978, only in 1995 the perfect solution for the problem was proven not to exist [14] (even though, by changing the formulation of the problem it can be solved [8]). In spite of that, a number of empirical and theoretical advances around the DCT have been achieved, and many techniques – mostly,

evolutionary computation based – have been developed that have continuously led to better and better rules, even though the best possible imperfect rule remains unknown. Strikingly, the DCT has been proven to be solvable by rule combinations, even the trivial combination of running elementary rule 184 for N (the lattice size) time steps, followed by elementary rule 232 [7] (a result then generalised in [5] and [4]).

Another very interesting case along the same line is concerned with the *Parity Problem*, by which the CA evolution has to go to a final configuration of all 0s, if the number of 0s in the initial condition is even, or to 1s, otherwise. Although it is still unknown whether there exists a single-rule solution to the problem (in fact, there are empirical evidences that there is not [10]), a solution to the parity problem has been found when 5 elementary rules are combined in sequence, in a certain way [3].

The combination of CA rules in sequence defines an immense phenomenological space that is completely unknown, with potential for extremely interesting theoretical consequences; for instance, one could ask what is the space of computable functions that is implicit in the elementary space, out of combinations of its null and fixed point rules 2 by 2, 3 by 3, and so on. This is virtually a new domain of enquiry for which no theoretical account is currently available, but that can be addressed from an empirical perspective. This is precisely what is carried out in this paper. Here, a genetic algorithm [6] is presented that leads to successful sequential combinations of up to 4 elementary rules that can solve the DCT. Then, by analysing the rules involved, various other successful combinations are derived.

In the next section the evolutionary algorithm is presented. Section 4 then discusses the experiments and analyses carried out, and then conclusions are drawn.

2 The Evolutionary Algorithm

Our algorithm is based on [1], where the same problem was tackled, with limited but encouraging results.

Basically, the role of the evolutionary algorithm is to search for a combination of n elementary CA rules and the corresponding number of time steps that each one is meant to run through. Therefore, each candidate solution (chromosome) can be represented by a sequence of genes, each one containing a rule and the number of time steps it is meant to iterate over its initial configuration (see Figure 4). Naturally, the initial configuration for R_1 is the actual binary sequence that defines the DCT, and those for the subsequent R_i rules are taken from the last configuration generated after running R_{i-1} for t_{i-1} time steps.

$\mathbf{R_1}$	t_1	$\mathbf{R_2}$	t_2	.....	$\mathbf{R_n}$	t_n

Fig. 4. Representation of a candidate solution: rule R_i is run for t_i iterations

As an illustration, Figure 5 displays the representation of a chromosome with 2 genes: rule 110 applied 30 times, and rule 137 applied 119 times. While in the actual algorithm the rules are coded in binary and the time steps in decimal, for the sake of simplicity decimal notation is used herein for both.

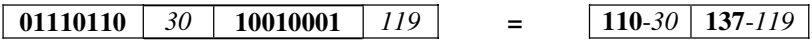


Fig. 5. Example of a chromosome with 2 genes and the decimal notation for rule and iteration

The fitness of every individual is simply the fraction of all N_{ICs} initial configurations in which the individual produced the correct final configuration (of all 0s or all 1s), when attempting to solve the DCT on them. The set of initial configurations (ICs) is generated with uniform distribution in respect to the amount of 1s in them. The lattice length is assumed to be an odd number so as to ensure that any initial configuration is valid for the task.

The following evolutionary algorithm is used and runs for N_{gen} generations:

- The population is composed of N_{pop} individuals, each one with N_r genes.
- A single set of N_{ICs} initial configurations of length N is generated and used throughout the evolutionary process for evaluating the candidate solutions.
- The top N_{elite} individuals at every generation are transferred directly to the next, without modification.
- Through (deterministic) tournament selection of size 2, involving the entire population, the remaining (N_{pop}-N_{elite}) individuals of the next generation are formed through the action of crossover followed by mutation.
- Every pair of selected individuals is subjected to a single-point crossover.
- Each offspring produced after crossover undergoes mutation, as follows:
  - With probability 0.25, one of the rules in its genes mutates, by the flipping one of the bits in its binary number representation.
  - With probability 0.25, one of the iterations appearing in its genes mutates, through the addition or subtraction of a random number, uniformly distributed in the interval $[-\lfloor N/(2\times N_r)\rfloor, +\lfloor N/(2\times N_r)\rfloor]$.
  - With probability 0.25, both the latter mutations happen, possibly involving two different genes.
  - Finally, with probability 0.25 the individual does not mutate.

The main differences between our work and the one described in [1] are: our algorithm relies on a completely distinct mutation scheme and, apparently, their use of the set of initial configurations is different from ours; our work is supplemented by analyses of the evolved rules, based upon properties of the elementary CAs; they only reported results with 2 genes, while we used up to 4; they found only 1 successful solution for DCT (although they reported 2), while we found 4, which, through the analyses hinted at above, eventually led to a total of 28 different rule combinations (24 3-rule solutions and 4 2-rule solutions).

3 Experiments and Analyses of the Results

Our first intent was to check whether our algorithm could also lead to the results reported in [1] for the DCT, which was successfully carried out. The main experiments were conducted under the following conditions: k=2, r=1, N=149, N_{ICs} =1000, N_{pop}=100, N_{elite} =20, N_r =2, and $t_1 + t_2 = 149$. Figure 6 compares all results.

KW$_1$:	184-*124*	232-*25*
KW$_2$:	226-*73*	232-*76*

MO$_1$:	184-*75*	23-*74*
MO$_2$:	184-*73*	23-*76*
MO$_3$:	184-*73*	232-*76*
MO$_4$:	226-*74*	232-*75*
MO$_5$:	226-*73*	23-*76*

Fig. 6. Comparison of the results for DCT with 2 rules: on the left those by [1] (KWi), and on the right our results (MOi)

Basically, the differences between the two sets of results are in the numbers of iterations of each rule. However, it can be proven that KW_1 is not a valid DCT solution, since its number of iterations is less than the necessary. We get back to this point later.

The same evolutionary mechanism was also used in a search for DCT solutions with individuals formed by 4 genes. The experiments were carried out under the same conditions as before, except that now $N_r = 4$ and $t_1 + t_2 + t_3 + t_4 = 149$.

Most trials finalised with good solutions but none with perfect score; however, after 698 generations, the combination **226**-*73* **236**-*1* **51**-*0* **128**-*75* was generated as a perfect solution (certified under 50000 initial conditions). Interestingly, notice that it is in fact equivalent to a 3-rule solution, since the third rule runs for 0 iterations.

Analysing Solutions With 2 Genes. First of all, let us point out that it has been shown that the set of possible configurations that can appear in the evolution of one-dimensional cellular automata, at any given finite number of time steps, from all possible initial conditions, can be described by finite automata, that is, the resulting limit set at every time step, is always a regular language [12]. Therefore, we can rely on the notation of regular expressions for representing the action of elementary rules.

Analysing the results in Figure 6, notice that one of the second rules (namely, rule 23) is novel in respect to any known results until then; and it can be seen that it is one of the few rules in the elementary space that does not have a dynamical equivalent.

Notice also that our third and fourth results are similar to those in [1], as they have rule 232 as the second rule in the combination, and rules 184 or 226 as the first rule. The combination 184-232 is exactly the one reported in [7], but the other association, 226-232, can be explained by the fact that rule 184 is dynamically equivalent to 226.

Rule 184 is the so-called traffic rule. It moves any 1 to the right in the lattice, if its right-hand neighbouring site is 0. Its equivalent rule, 226, does the same, but moving 1s leftwards, provided the site at the left is 0. Both arrange the lattice with alternating 0s and 1s, so that only the prevailing bit (the one that appears the most) can be found in consecutive sites. Considering any initial condition as a regular expression like $(0+1)^N$ where N is the lattice length, after $\lfloor N/2 \rfloor$ -1 iterations both rules transform any initial condition to a regular expression of the form $((01)^*1^+)^+$ if there is a predominance of 1s, to $((10)^*0^+)^+$ if there is a predominance of 0s, or to $(01)^{N/2}$ if there are as many 1s as 0s. Regardless the case, the same original density of the initial condition is preserved, since rule 184 is conservative.

After the lattice has been sorted out, rule 232 completes the solution, by converting the entire lattice to 0s or 1s, according to the prevailing density. After $\lfloor N/2 \rfloor$ iterations,

it transforms any initial condition of the form $((01)^*1^+)^+$ to 1^N, and those of the form $((10)^*0^+)^+$ to 0^N. Each rule iteration increases the density of the prevailing bit. Rule 232 does not modify lattice configurations represented by the expression $(10)^+ + (01)^+$.

Similarly, once the lattice is organised, rule 23 is also able to complete the solution, by turning the entire lattice from its original configuration to another, fully 0 or 1, according to the prevailing density; however, it requires an even number of iterations, because each rule iteration increases the difference between the density of both bits, but alternating the prevailing bit. The required number of iterations is $\lfloor N/2 \rfloor$ if $\lfloor N/2 \rfloor$ is even, or $\lceil N/2 \rceil$ if $\lfloor N/2 \rfloor$ is odd.

Figure 7 shows the space-time diagrams obtained after 149 iterations, from combining the rules mentioned above that solve the DCT. The initial condition 75 1s (blacks) followed by 74 0s (whites), displayed at the top, with time running downwards.

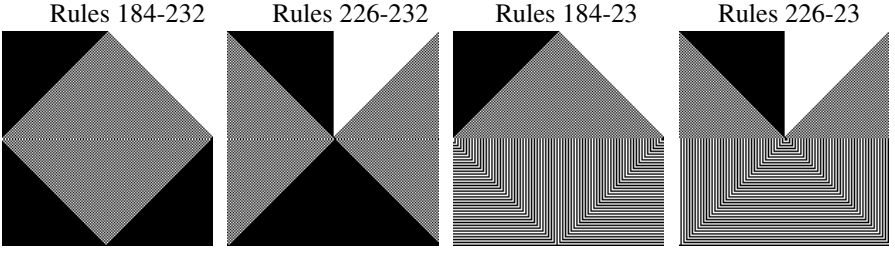


Fig. 7. Space-time diagrams with 149 iterations, of the 2-rule solutions to the DCT

Analysing Solutions With 3 Genes. Analysing the solution found with 3 rules notice that they have the same first rule as before. The difference is the exchange of the second rule by two new ones. By using the notion of dynamical equivalence it is possible to infer other solutions that also solve the problem, as shown below. Rule 200 is equivalent to rule 236 and rule 254 is equivalent to rule 128.

Found	**226**-*73*	**236**-*1*	**128**-*75*
Inferred	**184**-*73*	**236**-*1*	**128**-*75*
Inferred	**226**-*73*	**200**-*1*	**254**-*75*
Inferred	**184**-*73*	**200**-*1*	**254**-*75*

Fig. 8. Examples of 3-rule solutions to the DCT

After the lattice has been sorted out by rules 184 or 226, somewhere in it there must be a sequence of consecutive 1s or 0s. When rule 236 is applied just once, it can only preserve a 0 if there is another 0 beside it; the rest of the lattice turns to 1; so, if there are no consecutive 0s, the entire lattice turns to 1. In terms of regular expression representation, rule 236 transforms configurations of the form $1^*(01)^*0^K(10)^*1^*$, $K \geq 2$, to $1^*0^K1^*$. When rule 128 is applied it preserves a 1 only if both sides of the cell are also 1s; if there are consecutive 0s the entire lattice turns to 0. Rule 128 transforms configurations of the form 10^K1, $K \geq 2$, to 0^{K+2}, after $\lfloor N/2 \rfloor$ iterations.

Similarly, when rule 200 is applied just once, it keeps the state 1 only if there is another 1 beside it; the rest of the lattice turns to 0. If there are no consecutive 1s the entire lattice turns to state 0. Rule 200 transforms configurations of the form $0^*(01)^*1^K(01)^*0^*$, $K \geq 2$, to $0^*1^K0^*$. When rule 254 is applied it keeps the state 0 only if both sides of the cell are also 0s. If there are any consecutive 1s the entire lattice turns to 1 after $\lfloor N/2 \rfloor$ iterations. Finally, rule 254 transforms configurations of the form 01^K0 ($K \geq 2$) to 1^{K+2}.

The 3-rule solution does not solve the general formulation of DCT, in which the final configuration should have the form (01)* or (10)*, when the initial condition is balanced, that is, each bit is equally present in it.

With the same features of Figure 8, Figure 9 displays the space-time diagrams of the combinations involving the 3-rule solutions mentioned above.

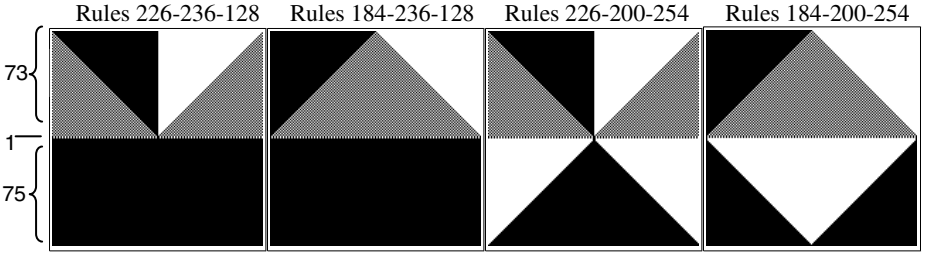


Fig. 9. Space-time diagrams with 149 iterations, of the 3-rule solutions to the DCT

Now, by observing the classification of the dynamical behaviour for the elementary rules and some of the values of the static parameters associated to their dynamical behaviour forecast (both mentioned earlier, at Section 1), additional 3-rule solutions can be inferred – as shown in Figure 10 – obtained by changing the third rule in the combination by others with the same dynamical regime.

Inferred	**226**-*73*	**236**-*1*	**160**-*75*
Inferred	**184**-*73*	**236**-*1*	**160**-*75*
Inferred	**226**-*73*	**200**-*1*	**250**-*75*
Inferred	**184**-*73*	**200**-*1*	**250**-*75*

Fig. 10. Other examples of 3-rule combinations that solve the DCT

More precisely, these solutions came out of the observation that rules 160 and 250 belong to the same dynamical class (namely, *null* behaviour) as rules 128 and 254, and that all of them share the same values for the absolute activity (A) for the activity propagation (P) parameters, respectively, the values of 0.25 and 0. But while the latter observation turned out to be positive towards inferring the new combinations, it was empirical in nature, thus calling for an effective explanation.

The search mechanism was also tested with other parameters. For instance, by not limiting the amount of iterations according to the lattice length, other solutions emerge, and others can be inferred, as shown finally, in Figure 11.

226-*73*	**236**-*1*	**136**-147
184-*73*	**236**-*1*	**136**-147
226-*73*	**236**-*1*	**168**-147
184-*73*	**236**-*1*	**168**-147
226-*73*	**200**-*1*	**234**-147
184-*73*	**200**-*1*	**234**-147
226-*73*	**200**-*1*	**238**-147
184-*73*	**200**-*1*	**238**-147

226-*73*	**236**-*1*	**192**-147
184-*73*	**236**-*1*	**192**-147
226-*73*	**236**-*1*	**224**-147
184-*73*	**236**-*1*	**224**-147
226-*73*	**200**-*1*	**248**-147
184-*73*	**200**-*1*	**248**-147
226-*73*	**200**-*1*	**252**-147
184-*73*	**200**-*1*	**252**-147

Fig. 11. Other 3-rule DCT solutions

4 Concluding Remarks

Performing computations with cellular automata, individually or arranged in space or time, opens up new conceptual issues in emergent, artificial life type forms of computation, and opens up the possibility of novel technological advances.

The exhaustive sequential combination of CA rules and their number of iterations can be an enormous combinatorial task. In order to transpose this obstacle evolutionary search has been shown herein as a definite possibility to be pursued. The GA managed to find solutions for the DCT, in particular in unexpected ways, as typically happens in natural and artificial adaptive systems.

However, the thrust of our findings is not due to the evolutionary search alone, as it was supplemented by analyses driven by background knowledge on dynamical behaviour of elementary rules and their parametric estimates, as well as those associated with the behaviour characterisation of the rules through the transformation in the regular expressions of the binary configurations involved.

The actual algorithm we used, despite its inspiration in [1], clearly surpassed it. The mutation process we defined, in addition to the analyses mentioned above were the key for the much better results we obtained. The fact that we represented the rules as a binary and the amount of iterations as a decimal number came from the scheme used in [1] and could well be an issue for further enquiry and possible improvement.

In respect to the efficacy of the solutions found by the GA, or inferred through analysis, they have been initially tested in ensembles of 50000 randomly generated initial conditions, before the rationale underlying the rule operations (formalised by the regular expression transformations) became apparent.

Other applications of the methodology described herein are currently under way, in particular in the parity problem, where our main objective is to simplify the only currently available solution to the problem, due to [3].

Acknowledgements

We thank MackPesquisa, the research funding programme of the Instituto Presbiteriano Mackenzie, for a research grant from Edital 2004, and Wolfram Research Inc for a Mathematica Academic Grant (No. 1149).

References

1. Kanoh and Y. Wu. “Evolutionary design of rule changing cellular automata”. In: V. Palade, R.J. Howlett, L.C. Jain (eds.). Knowledge-Based Intelligent Information and Engineering Systems, 7th International Conference (KES 2003, Oxford, UK, September 3-5, 2003), Proceedings, Part I. Lecture Notes in Computer Science 2773, Springer-Verlag, Available as www.kslab.is.tsukuba.ac.jp/~kanoh/kslab/study2/kanoh_KES2003.pdf, 258-264, 2003.
2. S. Wolfram. A New Kind of Science, Wolfram Media, 2002.
3. K.M. Lee, H. Xu and H.F. Chau. “Parity problem with a cellular automaton solution”. Physical Review E*,* 64:026702/1-026702/4, 2001.
4. H.F. Chau, L.W. Siu and K.K. Yan. “One dimensional n-ary density classification using two cellular automaton rules”. International Journal of Modern Physics C, 10(5):883-889, 1999.
5. H.F. Chau, K.K. Yan, K.Y. Wan and L.W. Siu. “Classifying rational densities using two one-dimensional cellular automata”. Physical Review E*,* 57(2):1367-1369, 1998.
6. M. Mitchell. An Introduction to Genetic Algorithms. Bradford Book, Reprint edition, 1998.
7. H. Fukś. “Solution of the density classification problem with two cellular automata rules”. Physics Review E, 55:2081R-2084R, 1997.
8. M. Sipper, M.S. Capcarrère and E. Ronald. “A simple cellular automaton that solves the density and ordering problems”. International Journal of Modern Physics *C*, 9(7):899-902, 1998.
9. M. Mitchell, J.P. Crutchfield and R. Das. “Evolving cellular automata to perform computations”. In: T. Back, D. Fogel and Z. Michalewicz (editors), Handbook of Evolutionary Computation*.* Oxford: Oxford University Press, 1998.
10. P.P.B. de Oliveira and R.B. Vaiano. “Searching for a cellular automaton to solve the parity problem” (in Portuguese). Unpublished manuscript under review, 2005.
11. G.M.B. Oliveira, P.P.B. de Oliveira and N. Omar, “Definition and applications of a five-parameter characterization of one-dimensional cellular rule space”. Artificial Life Journal, 7(3), MIT Press, p.277-301, 2001.
12. S. Wolfram. “Computation theory of cellular automata”. Communications in Mathematical Physics, 96:15-57, 1984.
13. W. Li. “Parameterizations of Cellular Automata Rule Space”. SFI Technical Report: Preprints, Santa Fe, NM, USA, 1991.
14. M.W.S. Land and R.K. Belew. “No two-state CA for density classification exists”. Physical Review Letters, 74(25):5148, 1995.
15. M. Mitchell. An Introduction to Genetic Algorithms. Bradford Book, Reprint edition, 1998.
16. S. Bandini, G. Mauri and R. Serra. “Cellular automata: From a theoretical paralell computational model to its application to complex systems”. Parallel Computing, 27:539-553, 2001.
17. M. Mitchell. “Computation in cellular automata: A selected review”. In: Nonstandard Computation. Weinheim: VCH Verlagsgesellschaft, 1996.
18. P. Sarkar. “A brief history of cellular automata”. ACM Computing Surveys, 32(1):80-107, 2000.

Penrose Life: Ash and Oscillators

Margaret Hill[1], Susan Stepney[1], and Francis Wan[2]

[1] Department of Computer Science, University of York,
Heslington, York, YO10 5DD, UK
susan@cs.york.ac.uk
[2] Ampleforth College, York, YO62 4ER, UK

Abstract. We compare the long term behaviour of Conway's Game of Life cellular automaton, from initial random configurations, on a bounded rectangular grid and a bounded Penrose tiling grid. We investigate the lifetime to stability, the final 'ash' density, and the number and period of final oscillators. Penrose grids have similar qualitative behaviour but different quantitative behaviour, with shorter lifetimes, lower ash densities, and higher ocurrence of long-period oscillators.

Keywords: Conway's Game of Life; Penrose tiles; ash; oscillators.

1 Introduction

John Horton Conway's *Game of Life* [1][3] is a simple two-dimensional, two state cellular automaton (CA), remarkable for its complex behaviour [1][8]. That behaviour is known to be very sensitive to a change in the CA *rules*. Here we investigate its sensitivity to changes in the *grid*, by the use of an aperiodic Penrose tiling grid [4][7].

2 Varieties of Life

In Conway's *Game of Life* CA, the neighbourhood of each cell comprises the 8 nearest cells of the Moore neighbourhood. Each cell has two states, 'dead' and 'alive'. If a cell is alive at time t, then it stays alive iff it has 2 or 3 live neighbours (otherwise it dies of 'loneliness' or 'overcrowding'). If a cell is dead at time t, then it becomes alive (is 'born') iff it has exactly 3 live neighbours.

Life has grown its own extensive and idiosyncratic terminology over the years. Much of this is collected in the extensive on-line *Life Lexicon* [9]. In particular, an initial random starting state is called a *soup*, and, mixing metaphors somewhat, the final resulting configuration is called the *ash*.

We can run CAs such as Life on aperiodic grids, such as Penrose tilings, with a suitable definition of the 'neighbourhood'.

There are two classic sets of Penrose tiles [4][7], *kites* and *darts* (so called because of their shapes) and fat and thin rhombuses. We use the kite and dart form. A plain kite and dart can be combined into a rhombus, and so tile the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 471–480, 2005.

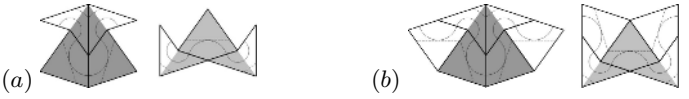


Fig. 1. Deflating a kite and a dart (a) minimal deflation; (b) deflation avoiding 'holes' in the result

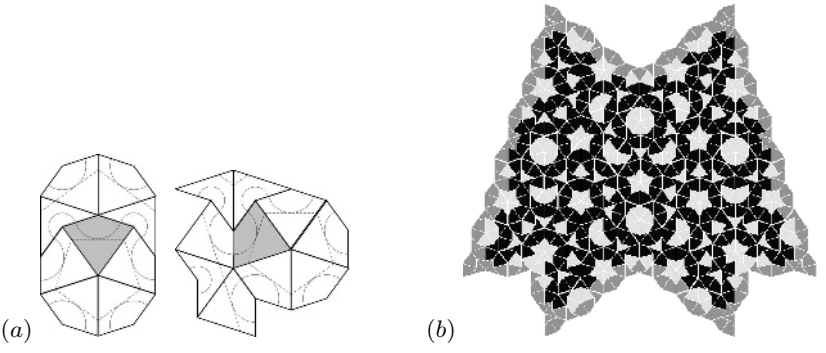


Fig. 2. Variable Penrose neighbourhood: (a) example of kites with eight and nine neighbours; (b) numbers of neighbours in a small deflated grid: light coloured tiles have 8 neighbours; dark tiles have 9 neighbours; grey tiles are edge tiles with fewer than 8 neighbours.

plane periodically. To force the tiling to be aperiodic, *matching rules*, marks on the tiles that must be matched together, are used.

A valid Penrose tiling has no gaps or overlapping tiles. The deflation algorithm [6] guarantees a valid tiling. At each round of deflation, each kite and dart tile is replaced with smaller kites and darts (figure 1). This leads to overlapping tiles, but the overlap is exact, and so the extra tiles can be safely removed. Since we use this deflation algorithm, we are restricted to the sizes (number of cells) of Penrose grids produced by the successive deflation generations.

In a rectangular grid, four cells meet at every vertex, and every cell has eight neighbours. In a Penrose grid, three, four or five cells can meet at a vertex, and Penrose grid cells can have either eight or nine neighbours (figure 2). We have found no algorithmic way of reducing the neighbourhood of all tiles to eight whilst maintaining the undirected nature of the neighbourhood graph. So we leave the neighbourhood as it is, and apply the Life rules to it unchanged.

We need to cope with the edge of the deflated Penrose grid. There are two conventional ways in CAs of removing the effect of the edge of the grid.

1. Periodic Boundary Conditions: the grid has the topology of a torus, finite but unbounded (has no edges). This is the approach usually taken for investigating statistical properties of soups, with the results more or less tentatively extrapolated to infinite grids. However, this approach is impossible for aperiodic grids such as a Penrose grid.

Table 1. The four grid sizes investigated: four Penrose deflations and the corresponding nearest regular square grid size.

	S, small	M, medium	L, large	X, extra-large
Penrose	688	1907	5170	13900
rectangular	$676 = 26^2$	$1936 = 44^2$	$5184 = 72^2$	$13924 = 118^2$

2. Lazy Infinite Grid: implemented by lazily expanding a finite grid as activity nears its edges [5]. This is the approach usually taken for investigating the properties of particular structures, such as glider guns. It is not practical for implementation on Penrose grids produced by deflation, since the generation n grid does not clearly appear as a subpart of the larger generation $n + 1$ grid.

Since neither of these standard approaches is suitable for investigating Penrose soups, we choose to investigate the effects of having a *bounded* grid, explicitly noting the effect of the edges. We have to decide how to handle the boundary. We can choose the border cells to stay 'dead', no matter what their neighbours' states, or choose them to have a reduced neighbourhood of five (or three at the corners). These choices are equivalent for CA rules like those of Life, where the state transition depends only on the *total* number of live neighbours (so permanently dead neighbours are equivalent to no neighbours).

3 Experimental Set-Up

For both the regular and Penrose grids, to compare like with like, we investigate the behaviour of a finite grid, initially empty except for a smaller patch of soup. Given that we are restricted to certain Penrose grid sizes by the deflation algorithm, we restrict the rectangular grid to the nearest similar sizes (table 1).

So our investigations are parameterised by the initial soup patch size S, initial soup density D, and fixed grid size G. The questions we pose are:

- what are the lifetimes of the initial random configurations?
- what are the final ash densities?
- what are the periods of the oscillators at the end of the lifetime?

A pattern is called a period n (pn) oscillator if it repeats after n generations. We define the *lifetime* of a configuration as the number of generations from the initial random starting configuration until it stabilises to ash, where a *stable* state comprises only oscillators, including 'Still Life' $p1$ oscillators. So we wait until any gliders have been absorbed by the boundary.

For example, on an infinite grid, the well-known 'r-pentomino' initial configuration reaches a stable state after 1103 generations, when it comprises several Still Lifes and $p2$ *blinkers*, and six escaped gliders. On a finite but large enough grid it stabilises to ash once the six gliders reach the boundary.

4 Lifetime Results

We use 1000 runs with each parameter set, but different random starting configurations.

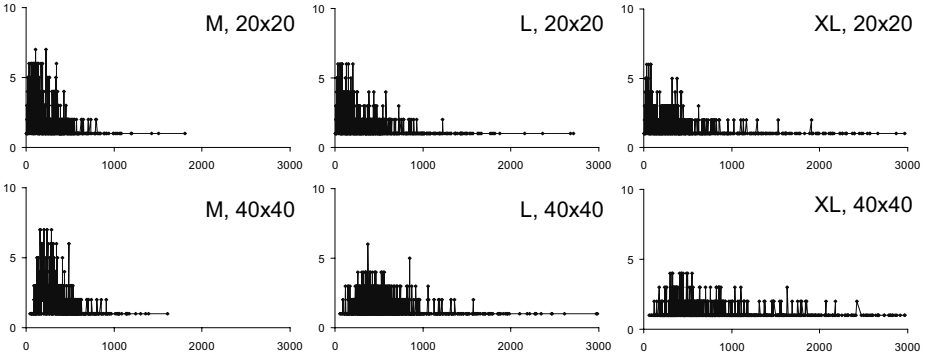


Fig. 3. Frequency against lifetime to stability on a regular grid, for initial density $D = 20\%$. Columns are increasing grid sizes G, showing lengthening lifetime tails; rows are increasing soup patch sizes S, peaks move to longer lifetimes.

4.1 Classic Life

Previous Results. Achim Flammenkamp has an extensive list of ash objects grown from soup [2]. He starts with initial densities $D = 0.371 - 0.375$, on toroidal (periodic boundary condition) grids of sizes $2^{12} \times 2^{12}$ (4096×4096) and $2^{14} \times 2^{14}$ (16384×16384). These experiments give an asymptotic ash density of 0.0287115 bits per cell. There are no reports of the time taken to stabilise.

The grids we test here are much smaller (the largest grid we test is X = 118×188), but we test a much broader range of grid sizes and initial densities.

Distribution. The distributions of lifetimes are highly skewed (figure 3). There is a peak in the distributions at low lifetimes, with a long tail of high lifetimes. This tail is longer on larger grids, that is, larger grids can support longer lifetime structures, implying that there is a correlation between a structure's lifetime and its size. The peak lifetime is higher with larger initial patches: small patches tend to die more rapidly than larger patches.

Effect of Initial Density. We expect a short lifetime at low density, because there are too few live cells to cause more to be born. We might also expect short lifetimes at high density, as everything dies of overcrowding.

We can see these effects qualitatively from the following approximate argument. (It is only an approximation to the real behaviour, because we treat the distribution as smooth, yet clumpiness has an important effect; however, the intuition it provides is sound.) Let the average density at time t be ρ_t. Then the average density at time $t + 1$ will be (approximately):

$$\rho_{t+1} = \rho_t P(\text{staying alive}) + (1 - \rho_t) P(\text{being born}) \tag{1}$$

$$= \rho_t \left(P(2 \text{ nbrs}) + P(3 \text{ nbrs}) \right) + (1 - \rho_t) P(3 \text{ nbrs}) \tag{2}$$

$$= \rho_t \left({}_nC_2 \, \rho_t^2 (1 - \rho_t)^{n-2} \right) + {}_nC_3 \, \rho_t^3 (1 - \rho_t)^{n-3} \tag{3}$$

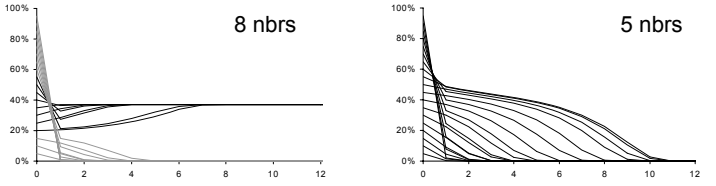


Fig. 4. Density against timestep for smoothed evolution of initial densities

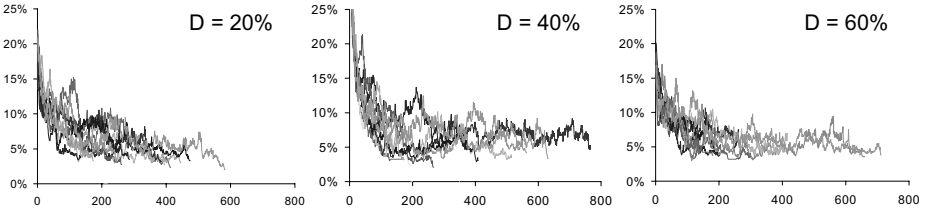


Fig. 5. Density against time on an M grid, of 10 initial random densities D

where $P(m \text{ nbrs})$ is the probability that m neighbours are alive. For a neighbourhood of $n = 8$, we get

$$\rho_{t+1} = 28\rho_t^3(1-\rho_t)^5(3-\rho_t) \tag{4}$$

For an initial ρ that is not too large or too small, the density rapidly converges to $\rho_\infty \approx 37\%$ (figure 4a). However, for low initial densities ($\rho_0 \lesssim 20\%$), there is not enough activity to sustain Life, and the density rapidly falls to zero. For high initial densities ($60\% \lesssim \rho_0$), there is massive death in the first generation, and the resulting density ρ_1 is less than the critical value, and so again rapidly converges to zero.

Similarly, for a neighbourhood of $n = 9$ (relevant to some cells in the Penrose grid), we get

$$\rho_{t+1} = 12\rho_t^3(1-\rho_t)^6(10-3\rho_t) \tag{5}$$

For an initial ρ not too large or too small, the density rapidly converges to $\rho_\infty \approx 35\%$.

We compare these calculations with actual runs, on an M grid (44×44 cells), with initial densities of 20%, 40%, and 60% initially covering the entire grid, for 10 runs each. The evolutions of the densities are shown in figure 5. We see that the evolution is qualitatively similar, but the actual densities are lower than the calculation, at closer to 10% whilst still evolving, and 2–5% once stabilised to ash. (This latter figure is consistent with Flammenkamp's asymptotic value of $\approx 2.9\%$.) The lower densities demonstrate the importance of clumpiness. There is no correlation between lifetime to stability, and ash density (figure 8a).

Figure 6 shows graphs of mean lifetime against density, for various grid sizes and initial patch sizes. At low densities, they have low lifetimes, and as the density rises, so does the lifetime. Once the density gets too high, lifetimes start

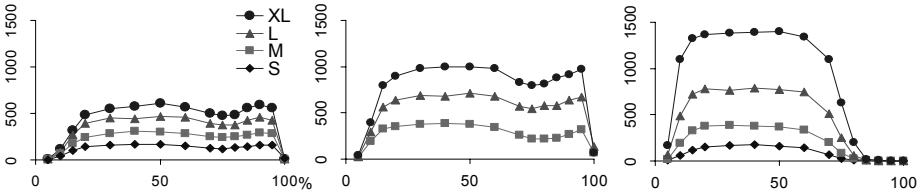


Fig. 6. Mean lifetimes to stability against initial density, on a regular grid, for different soup patch sizes $S = 20 \times 20$, 40×40, full grid

to drop again. But then, for very high densities, the mean lifetime starts to rise once more (except when the initial patch completely fills the grid). Why?

These experiments are run on patches of initial random soup that are smaller than the total grid size. For very high densities, this initial patch is essentially a solid square. Note that the average density at the edge of a patch is half the interior density. So for solid patches, their edges are at a density suitable for sustaining Life. The centre rapidly dies, but the edges survive and propagate for a long time. So the entire graph can be thought of as having two components: one due to the central region, peaking near $D_{max} \approx 50\%$, and one due to the edges, peaking at $\approx 2D_{max}$.

Detailed Effect of Patch and Grid Size. The earlier figure 3 shows that both average and maximum lifetimes increase with initial soup patch size S, and grid size G.

The bigger the grid, the longer the lifetime: the boundary does seem to be 'killing' the life. Clearly, if some central region shoots out gliders, then bigger grids will give longer lifetimes, because the lifetime is taken once all the gliders have hit the boundary, which will take longer for larger grids. But ours are all relatively small grids, and that is not the dominant effect: the soup is 'boiling' over the whole grid.

We can see that 5 neighbours (the case on the boundary) is not enough to maintain Life, using equation 3 for $n = 5$ neighbours. We get

$$\rho_{t+1} = 10\rho_t^3(1 - \rho_t)^2(2 - \rho_t) \tag{6}$$

In this case, whatever the initial value of ρ, it quickly converges to zero (figure 4b): all Life dies.

4.2 Penrose Grid Lifetimes

Now that we understand the effect of grid size, patch size, and initial density on the regular grid, we can investigate the effect of using an aperiodic Penrose grid.

Distribution. The lifetime distributions show the same qualitative behaviour as for a regular grid: the skewed distribution, and the increasing lifetimes with grid and patch size. However, the mean lifetimes to reach stability are approximately an order of magnitude *smaller* (figure 7).

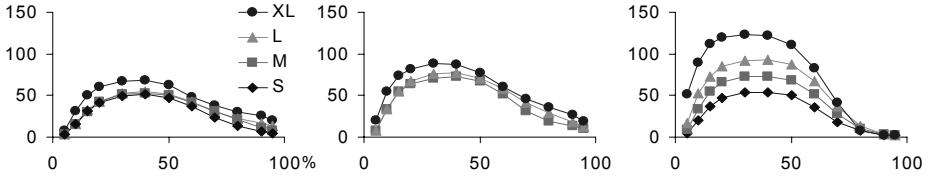


Fig. 7. Mean lifetimes to stability against initial density, on a Penrose grid, for different soup patch sizes $A = 400$, 16040, full grid

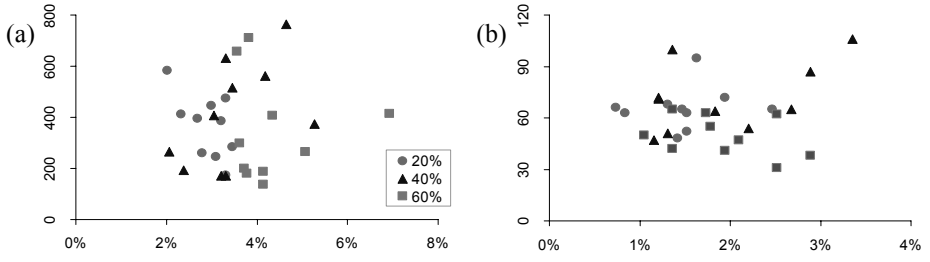


Fig. 8. lifetime to stability against ash density, for initial densities $D = 20\%$, 40%, 60%, on M size grids: (a) rectangular grid (b) Penrose grid.

The densities drop much faster than they do on the rectangular grid, last for a much shorter time at the low density before stabilising, and produce lower density ash, 1–3% (figure 8b).

5 Oscillator Distribution

It appears that the aperiodic grid stops structures propagating any distance and affecting distant objects. So everything becomes 'Still P-Life', or an oscillator, much sooner. We investigate this further, by looking at the distribution of oscillator periods in the ash.

5.1 Life Oscillators

The Flammenkamp web site [2] has an extensive list of oscillators grown from soup. The vast majority are $p2$. Why do we find so very few $p3$ and higher oscillators in the ash?

There are many kinds of $p2$ oscillators. The *blinker* has only three active cells; there are also three 6 cell $p2$ oscillators (*beacon*, *clock*, and *toad*). These readily form by chance in the ash. Similarly, the commonly-occurring *glider* has only five active cells.

The smallest $p3$ oscillator, the *caterer* (discovered by Dean Hickerson in 1989), however, has 12 cells active in its smallest configuration. This is much

less likely to occur by chance than smaller period oscillators, and so is unlikely to be found in the ash. Our tests never discovered a caterer oscillator.

The $p3$ oscillator that does occur in our tests is the *pulsar*, which has 24 cells active in its smallest configuration. Surely this is even *less* likely? However, the pulsar has a 10 state predecessor, which is more likely to occur than the caterer.

The smallest $p4$ oscillators (*mazing* and *mold*) have 12 cells active in their smallest configuration, and the smallest $p6$ oscillator (*unix*) has 16 cells, so these are also unlikely to occur by chance, unless they have small state progenitors, too. They never occurred in our tests.

5.2 Penrose Oscillators

Although the Penrose grid is aperiodic, any Penrose oscillator is of general interest, because a Penrose tiling has the *recurrence property*: any given finite patch of Penrose tiling recurs in infinitely many other patches, in any Penrose tiling. (This does not conflict with what we said earlier about being unable to find the deflated generation n grid within the generation $n + 1$ grid: these grids are finite, whereas the recurrence property applies to full, infinite, tilings.) Hence a given oscillator confined to a particular patch of a given tiling can also occur in infinitely many other patches, in any Penrose tiling.

This property also means that we are justified in confining our experiments to just those Penrose grids obtained by deflating a single dart: any finite patch that occurs in any Penrose tiling will occur in our grids (provided that they are big enough, of course).

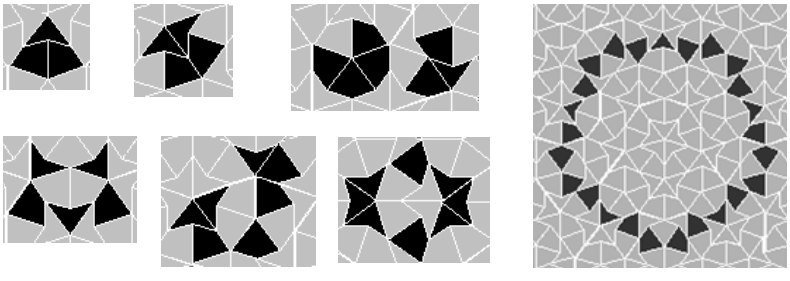


Fig. 9. Examples of Penrose still lifes. All the small examples were found in the ash; the large ring was hand constructed.

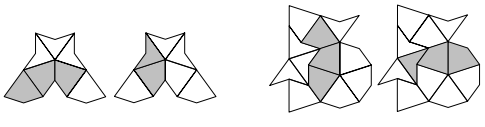


Fig. 10. Two of the $p2$ Penrose oscillators, named *plinker*s by analogy to the Life three cell *blinker*

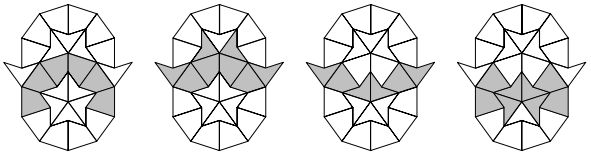


Fig. 11. A 6 cell $p4$ Penrose oscillator, the *bat*

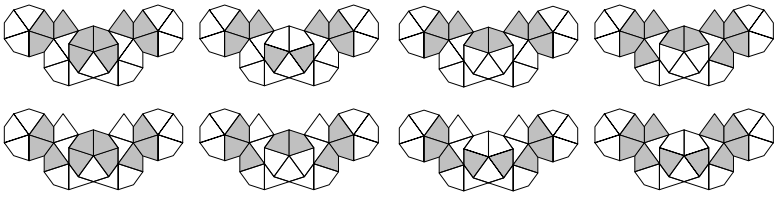


Fig. 12. An 8 cell $p8$ Penrose oscillator (read across the rows, then down the columns)

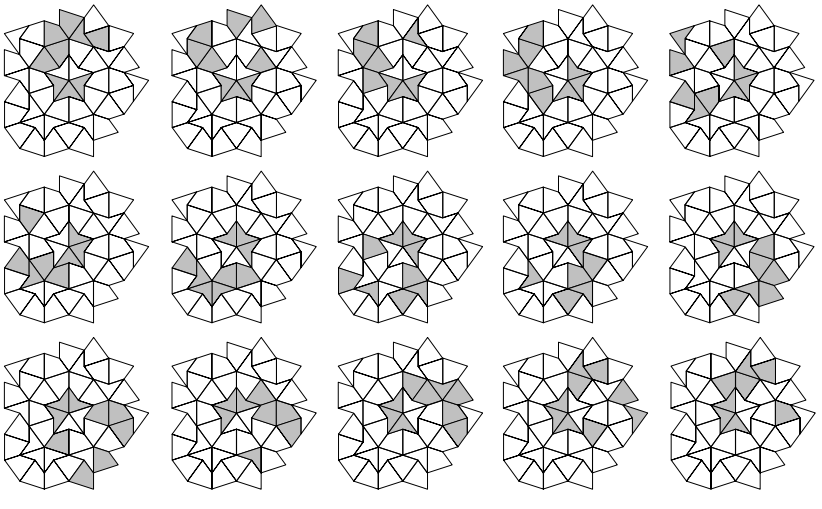


Fig. 13. An 8 cell $p15$ Penrose oscillator, the *dancer* (read across the rows, then down the columns)

We have discovered a rich zoo of small-period P-Life oscillators in the ash. There are many $p1$ still lifes (a few examples are shown in figure 9), including one with three cells, some forming closed loops, and some disconnected.

Because of the two different cell shapes, and the different neighbourhood sizes, many oscillators come in several variants. For example, the ash contained *six* different three-cell $p2$ oscillators, analogous to the single *blinker* (figure 10).

Some of these visually distinct variants look more similar when considering just the neighbourhood topology, rather than the different tile shapes.

There is a 6 cell $p4$ oscillator that we dub the *bat* (figure 11); the ash exhibited four variant bats. The ash also threw up a symmetrical 8 cell $p8$ oscillator (figure 12).

Perhaps most amazingly, we discovered an 8 cell $p15$ oscillator (figure 13). When animated it appears that the live cells are 'dancing' around the central star formation. This movement is reminiscent of the behaviour of a glider on the regular grid: the $p15$ *dancer* is moving, but is confined to perpetually move in a circle because of the nature of the Penrose grid. We found no long range propagating structures, because of the aperiodic nature of the grid. However, the possibility of arbitrarily large rings (figure 9) hints at the possibility of arbitrarily large period dancers around such rings: such will almost certainly need to be hand constructed.

These relatively long period Penrose oscillators are more common than their regular Life equivalents, because they are so small. This still raises the question: why are smaller oscillators possible? The occasionally larger neighbourhood is a possible contributor. It certainly allows constructs such as arbitrarily large rings. Future work will investigate oscillator distributions, and the precise effect of the extra neighbour.

6 Conclusions

Life on a Penrose grid has similar qualitative behaviour to regular Life, but different quantitative behaviour. The lifetime to stability is an order of magnitude shorter, the ash density is about half, and there are more spatially small long-period oscillators.

References

1. Elwyn R. Berlekamp, John Horton Conway, and Richard K. Guy. *Winning Ways for Your Mathematical Plays Volume 2: games in particular*. Academic Press, 1982.
2. Achim Flammenkamp. Achim's game of life page. `http://wwwhomes.uni-bielefeld.de/achim/gol.html`, 2004.
3. Martin Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223(4):120–123, October 1970.
4. Martin Gardner. Mathematical games: extraordinary non-periodic tiling that enriches the theory of tiles. *Scientific American*, 236(1):110–121, January 1977.
5. R. Wm. Gosper. Exploiting regularities in large cellular spaces. *Physica D*, 10:75–80, 1984.
6. B. Grünbaum and G. C. Shephard. *Tilings and Patterns*. W. H. Freeman, 1987.
7. Roger Penrose. Pentaplexity. *Eureka*, 39:16–32, 1978.
8. Paul Rendell. Turing Universaility of the Game of Life. In Andrew Adamatzky, editor, *Collision-Based Computing*. Springer, 2002.
9. Steven Silver. Life lexicon, release 24. `http://www.argentum.freeserve.co.uk/lex.htm`, February 2005.

Playing a 3D Game of Life in an Interactive Virtual Sandbox

Daisuke Ogihara and Hiroki Sayama

Dept. of Human Communication,
University of Electro-Communications, Japan
{ogihara, sayama}@cx.hc.uec.ac.jp

Abstract. We propose a novel artificial-life-oriented media art "RomperSand", which applies a three-dimensional version of the Game of Life (GoL) CA for the construction of an interactive virtual playground. In RomperSand, two distinct sets of state-transition rules are combined together: one for simulating physically plausible motion of virtual sand particles and the other for realizing the GoL-like dynamic behavior of *living* structures. Players can operate several virtual tools to create, destroy, and interact with these structures. The system was implemented as a Windows application and was tested by several users, gaining positive appreciations from them.

1 Introduction

The creation of interactive art has been one of the most effective applications of Artificial Life studies [1,2]. Recently, such interactive art based on cellular automata (CA) dynamics have been proposed [3,4,5]. CA are easy to implement and parallelize, and are capable of generating very complex dynamics; hence they are potentially very useful as a means to generate unexpected complex patterns for media art. Because the influence of a user's interaction on the generated patterns is way too complex to understand or predict, however, it often results in defeating the user's motivation to keep interacting with the CA. This can be a crucial problem when CA are applied to interactive media art.

To address this problem, we assume that introducing an easily understandable, explicit "motif" would help users develop a concrete image of the work in mind and keep their desire to interact with it. Based on this idea, we have developed a novel artificial-life-oriented media art, RomperSand, adopting a sandbox as its motif. In RomperSand, two distinct sets of state-transition rules are combined together: one for simulating physically plausible motion of virtual sand particles and the other for realizing the Game-of-Life (GoL) [6,7] -like dynamic behavior of *living* structures. Players can operate several virtual tools to create, destroy, and interact with these structures.

In what follows, the concept and details of RomperSand are described, with some exemplar screenshots and comments from test players also reported.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 481–490, 2005.

2 System

2.1 Concept

The motif underlying RomperSand is a sandbox. This choice was taken based on the fact that almost all of us share a happy experience in early childhoods to play in a sandbox and enjoy our first interaction with natural/physical materials there. We expected that using a sandbox as the motif of our work would project onto it an easily understandable, familiar image of such joyful experiences with the physical world, thereby helping a user to maintain his/her motivation to keep interacting with complex media art.

RomperSand consists of a three-dimensional virtual arena, in which players can play with virtual sands as they want, using familiar tools such as a rake, a shovel and a water can as an interface.

In this artificial world, water plays a very important role to connect between *living* and *non-living*. Virtual sands are normally dry (i.e., *non-living*) and obey a default state-transition rule for the simulation of physically plausible motion of sands, such as free falls and avalanches. However, players can pour water onto sands using the water can tool. Once the sands become wet, they turn alive, i.e., they begin to behave following a completely different GoL-like state-transition rule. This change often creates unexpected, fancy dynamic structures flourishing out of the spot where water was dropped. In the meantime, the moisture absorbed in sands gradually evaporate. When they become completely dry again, they come back to under the governance by the inorganic virtual physics laws, being gravitated to the ground.

According to the concept described above, all living structures in this world are destined to die eventually because of the inevitable evaporation of moisture. This, however, is expected to motivate players toward continuing interacting with the world, hoping to see life revive and reflourish out of their hands and water cans.

2.2 CA Rules

Technically, RomperSand is a three-dimensional CA lattice space (size: $100 \times 100 \times 20$). Each cube takes one of the five states: *Empty*, *Sand*, *Water*, *Wet Sand*, and *Block* (not mentioned in this paper). In addition, *Water* and *Wet Sand* have an integer value as their internal states (ranging from 0 to 10000 for *Water* and from 0 to 100 for *Wet Sand*), which specifies the amount of water or moisture that the cube contains.

As mentioned in the previous section, RomperSand has two distinct state-transition rule sets. Switching between the two rules are triggered by the addition of water and takes place locally (i.e., there is no global switching). Details of each rule set are described below.

Default Rule. The default rule set works as pseudo-physics laws in this artificial world, where mass (or the total number of sands) is strictly conserved. To achieve such physical plausibility including conservation, this rule is implemented using

three-dimensional "Margolus neighborhoods" [8,9], i.e., any $2 \times 2 \times 2$-sized three-dimensional pattern in the space is mapped to another pattern of the same size by this rule. The designed physics laws are summarized as follows: *Sand* and *Water* fall in the air. *Sand* can stack onto other *Sand* cubes, but steep slopes are subject to avalanches. *Water* spreads when it reaches on the ground. More details of the default rule are as follows:

- Vertical falling of sand.
  When a top cube is *Sand* and a bottom cube is *Empty*, the states of top and bottom cubes are exchanged. By this rule, the sand particle falls vertically.
- Avalanche.
  When both the top and bottom cubes are *Sand* but there is an *Empty* cube next to the bottom cube, the top *Sand* is moved diagonally into that empty cube. This rule realizes avalanches of sands slipping down on slopes, potentially capable of simulating sandpile experiments [10] if many *Sand* cubes are dropped at the same location.
- Vertical falling of water.
  This rule is the same as that of *Sand*. When a top cube is *Water* and a bottom cube is *Empty*, the states of top and bottom cubes are exchanged. By this rule, the water drop falls vertically.
- Absorption of moisture.
  When the *Sand* cube comes in contact with a *Water* cube, the *Sand* cube changes its state to *Wet Sand*.
- Diffusion of moisture.
  A *Sand* cube is changed to a *Wet Sand* cube if there is a *Wet Sand* cube in its neighborhoods. This realizes the diffusion of moisture from sand to sand.

Moreover, each cube has a flag that indicates whether it receives an upward force or not. This force is originated from the bottom plane of the 3D lattice space and propagates through sand and water. If this flag is on, the cube does not obey the rule for vertical falling.

- Spread of water.
  A *Water* cube whose flag is on changes the state of its neighbor *Empty* cube to *Water*. This rule is to make water that dropped down to the ground spread over the surface of the ground.

Figure 1 gives a schematic illustration of the default rule described above. Multiple state-transitions may apply in a single neighborhood at the same time if they are not obstructing each other. Figure 2 shows the flowchart of how the default rule determines the behavior of each *Sand*/*Water* particles. There are more technical details in the specific implementation of this rule, especially in terms of how the integer values specifying the amount of water/moisture in *Water*/*Wet Sand* cubes change in diffusion processes. These details are planned to be published elsewhere [11].

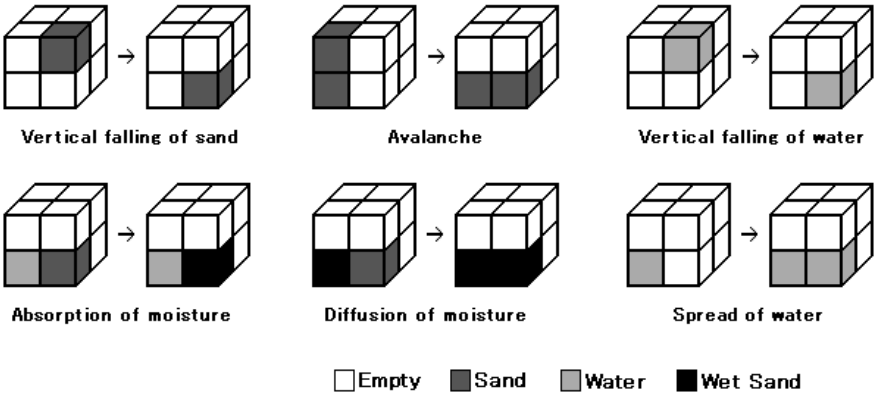


Fig. 1. Schematic illustration of the default rule

GoL-like Rule. When *Water* and *Sand* come to contact, the *Sand* cube becomes *Wet Sand* (i.e., *living*). Then, on this cube and its surroundings, the other GoL-like rule starts to apply, and lasts until they become completely dry again.

The size of its neighborhood template is 27 ($3 \times 3 \times 3$) with this rule. A new *Wet Sand* cube will appear if the total number of living cubes in the surrounding neighborhoods is either 4 or 5. A living cube will die if the number is not exactly 5. When cube changes from living to non-living, its state will be cleared to *Empty*. In our implementation, however, we heuristically set the probability of this type of state-transitions to be 1/2, in order to make the generated behavior more interesting.

2.3 Implementation

RomperSand was implemented as an application software that runs on a Windows (2000 or XP) based PC. It was written in C++ using Direct X. It can be downloaded from the authors' website[1].

Figure 3 shows a sample screenshot of RomperSand. A rake, a water can, and a shovel are displayed as icons at the upper left corner of the window. Players can select which tool to use by clicking on these icons. The rake can be used to collect sands. The water can is used to sprinkle water. The shovel can be used to dig up and scatter sands. Figure 4 shows exemplar scenes of the use of these virtual tools.

3 Characteristic Dynamics

RomperSand has been demonstrated internally in our facility, where more than 10 test players enjoyed playing with this artwork (Fig. 5). The players reported

[1] http://cx.hc.uec.ac.jp:8100/~ogihara/

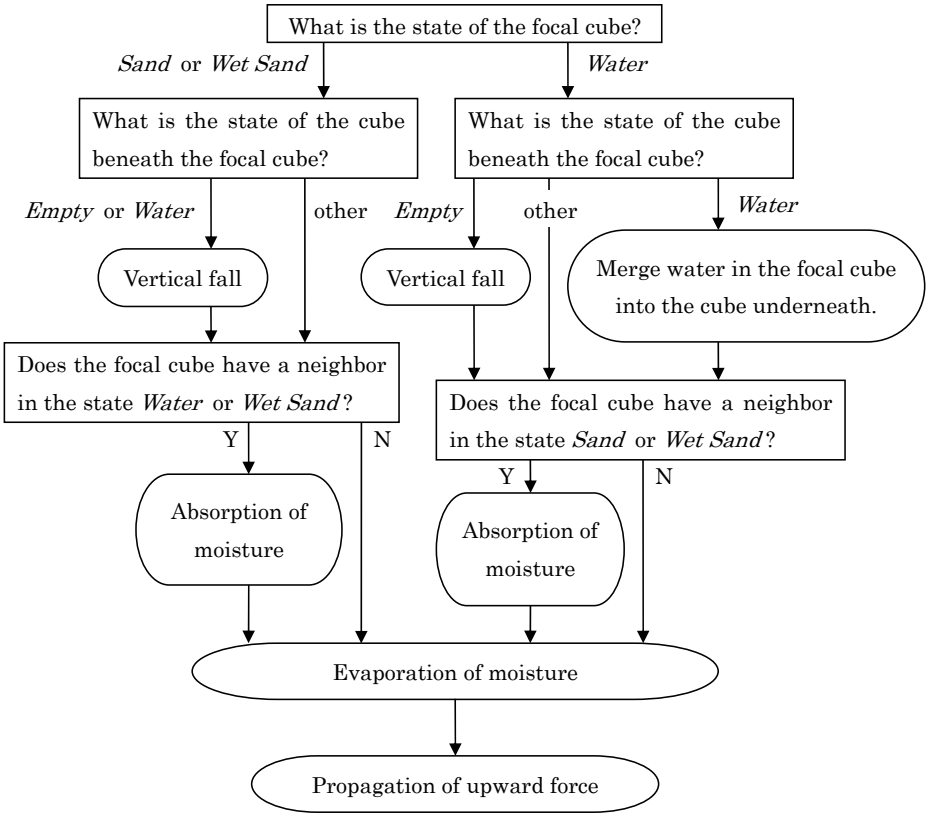


Fig. 2. Flowchart of the default rule

that there are at least three salient dynamic structures made of living sands that are easily identifiable in this world.

- Fountain of sands.
  A pattern in which sands come out one after another from the ground. This looks like a fountain of sands (Fig. 6).
- Elevating sands.
  A pattern of rising *Wet Sand* cubes. This pattern eventually ends up by scattering dried *Sand* cubes falling back to the ground, whose appearance is really dramatic (Fig. 7). This is considered similar to gliders in the original GoL.
- Crawling sands.
  A pattern that crawls toward one direction on the surface of the ground (Fig. 8). This is another kind of glider-like structure in this world. This is often mistakenly understood by the players as if there were some sort of living thing like a mole in action.

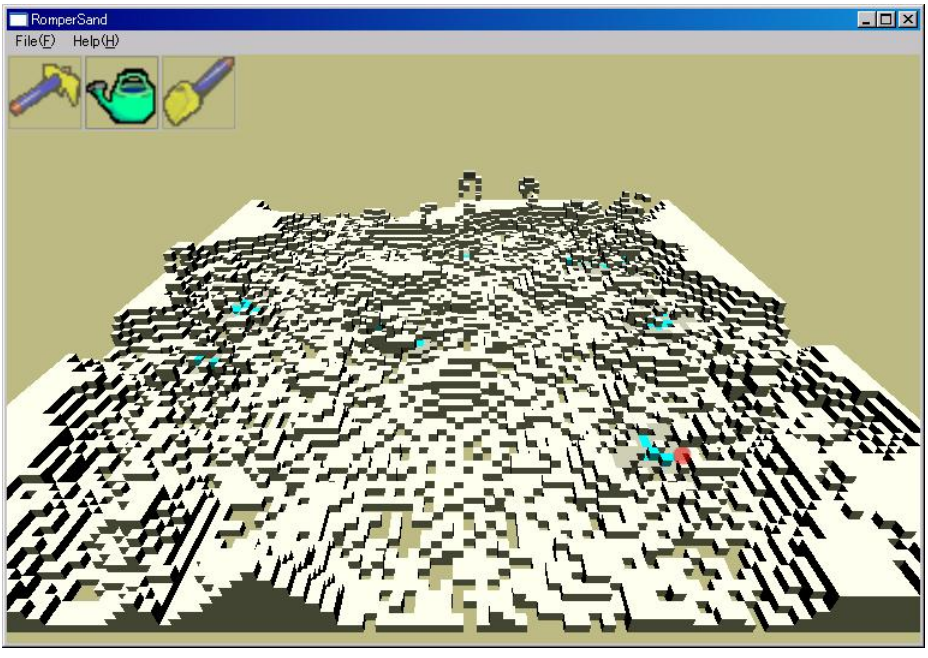


Fig. 3. A screenshot of RomperSand

Comments for RomperSand given by the test players are summarized below:

- "It is interesting that sand collapses when water is poured."
- "I think it is quite new that sands do irregular movements and splash sometimes."
- "The movement of sand is so drastic when I add water, so it is difficult to predict what I am going to make. But, because the collapsing scene was interesting, it is not frustrating."
- "This is very interesting to see as it moves by itself."

The duration of free play with RomperSand by the test players ranged from a couple of minutes to more than half an hour, about five minutes on average. All the test players appreciated the concept of this artwork and agreed with our intention to use a sandbox as its motif to promote users' motivation to interact with media art, suggesting that our original goal was achieved in this work to some extent.

4 Conclusion

We have presented an interactive artwork, named RomperSand, which includes as part of its dynamics a 3D version of the Game of Life. It aims to introduce a concrete motif of "sandbox" to CA-based media art that would otherwise be

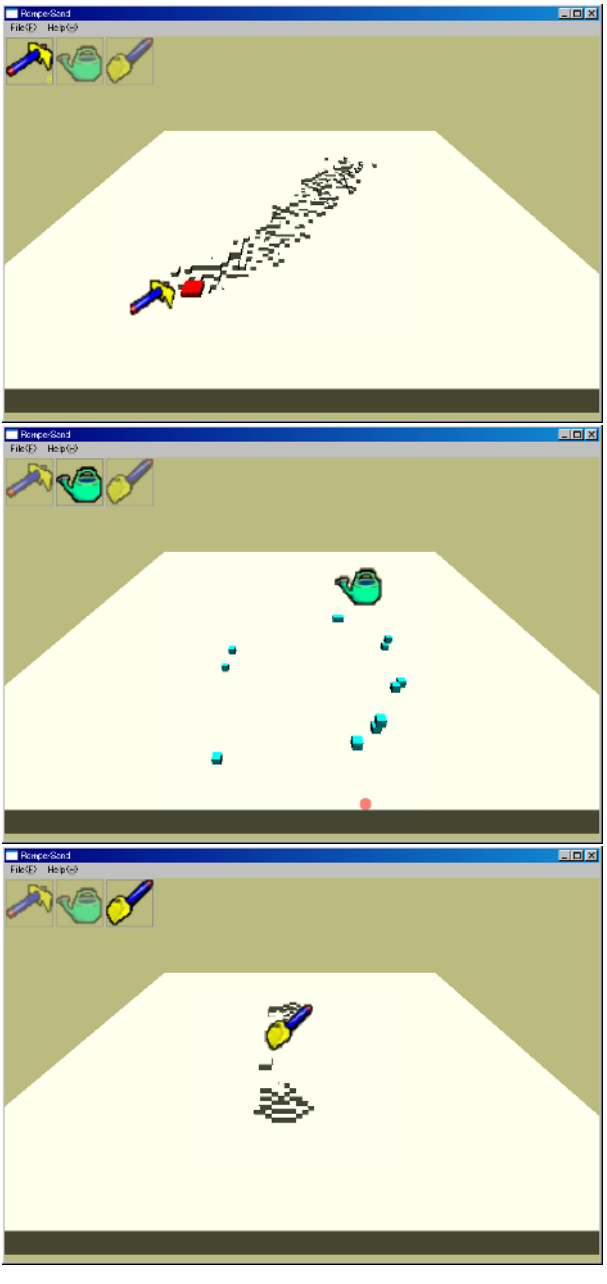


Fig. 4. Virtual tools in use (top: rake, middle: water can, bottom: shovel)

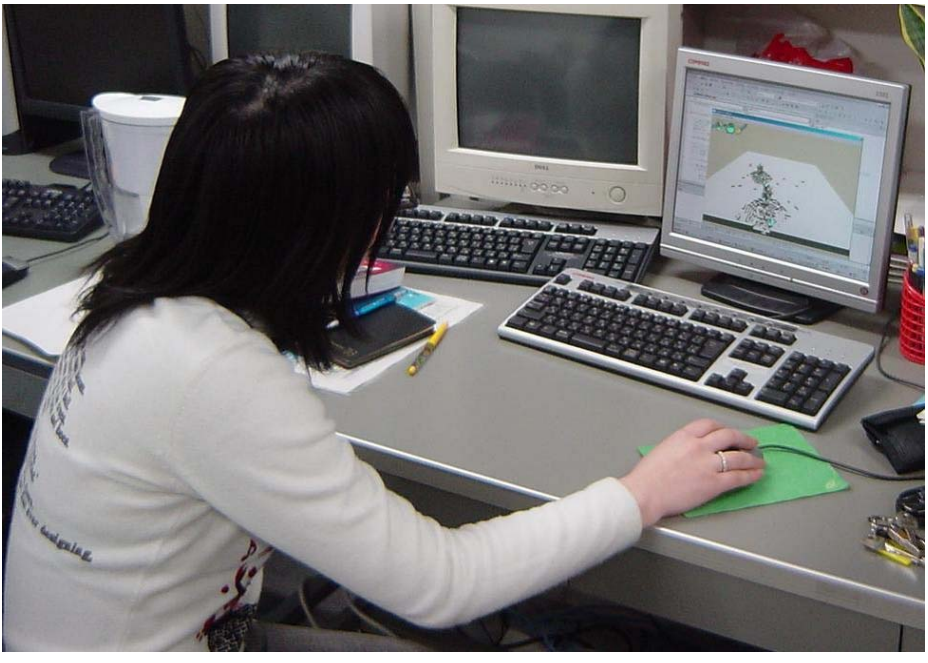


Fig. 5. A player playing with RomperSand

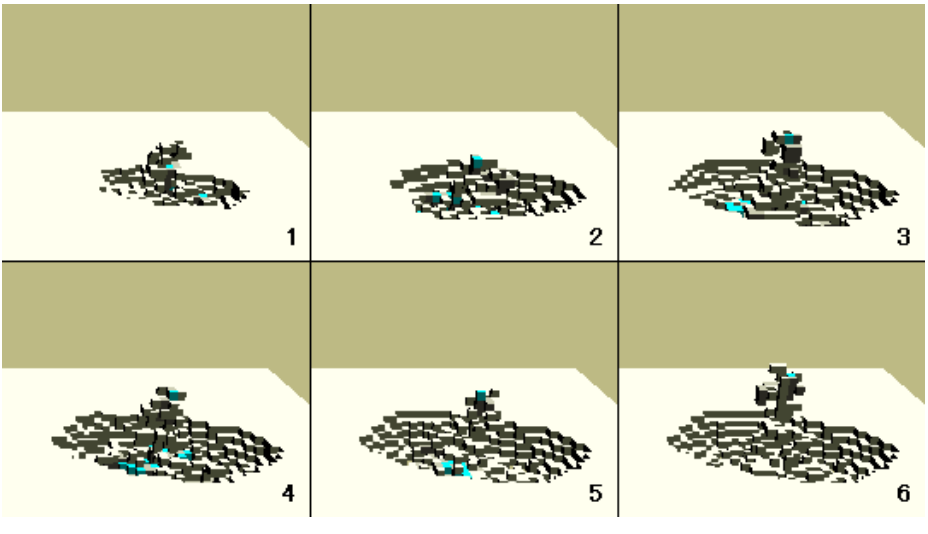


Fig. 6. Fountain of sands

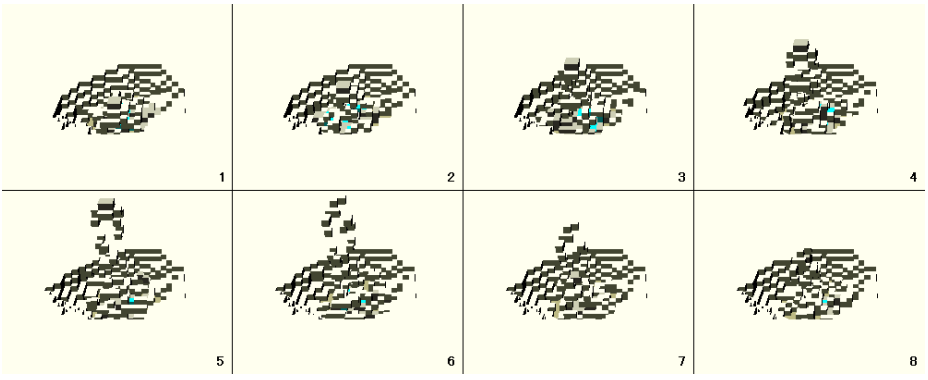


Fig. 7. Elevating sands

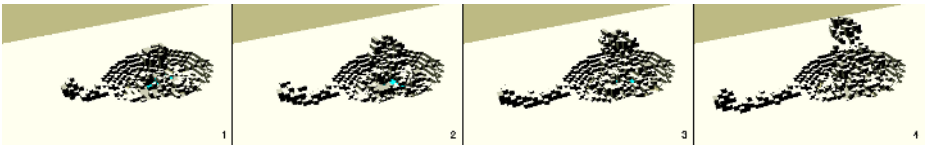


Fig. 8. Crawling sands

too complex to maintain the users' motivation for interaction. The combination of two distinct rules (the default pseudo-physics rule and the 3D GoL-like rule) was done by placing water onto the intersection between these two *non-living* and *living* regimes. Our attempt seems successful in view of positive reactions and appreciations received from the test players who played it.

RomperSand is still under major revision and development. Further improvement would be definitely needed, firstly for expansion of space (or increase of resolution of space) and secondly for increasing the speed of the CA simulation and response to the use of virtual tools. These would be crucial in improving the interactivity of RomperSand. A systematic analysis of the model's basic dynamics from technical/theoretical viewpoints would be another important direction of future work.

References

1. Sommerer, C. and Mignonneau, L.: Modeling emergence of complexity: The application of complex system and origin of life theory to interactive art on the Internet. In Artificial Life VII: Proceedings of the Seventh International Conference on Artificial Life, Bedau, M. A., McCaskill, J. S., Packard, N. H. and Rasmussen, S., eds., Cambridge, MA: MIT Press, 2000, pp. 547-554.
2. Mignonneau, L. and Sommerer, C.: Creating artificial life for interactive art and entertainment. In Artificial Life VII: Workshop Proceedings, Maley, C. C. and Boudreau, E., eds., 2000, pp. 149-153.

3. Dorin, A.: LiquiPrism: Generating polyrhythms with cellular automata. In Proceedings of the 2002 International Conference on Auditory Display, Nakatsu, R. and Kawahara, H., eds., Advanced Telecommunications Research International (ATR), Kyoto, Japan, 2002, pp. 447-451.
4. Tempesti, G. and Teuscher, C.: Biology goes digital: An array of 5,700 Spartan FPGAs brings the BioWall to "life". XCell Journal, Fall 2003, pp. 40-45.
5. Kodama, S., Fukuda, Y., Sayama, H. and Koike, Y.: Living Surfaces - A concept of animated texture using "Evoloop" and experiments in an interactive art. Journal of the Society for Art and Science 3: 193-196, 2004. Witten in Japanese.
6. Berlekamp, E., Conway, J. H. and Guy, R.: Winning Ways, vol 2, New York: Academic Press, 1982.
7. PoundStone, W.: The Recursive Universe. New York: William Morrow and Company, 1985.
8. Toffoli, T. and Margolus, N.: Cellular Automata Machines, Cambridge, MA: MIT Press, 1987.
9. Margolus, N.: Physics-like models of computation. Physica D 10: 81-95, 1984.
10. Bak, P., Chao, T. and Kurth, W.: Self-organized criticality. Phys. Rev. A 38: 364-374, 1988.
11. Ogihara, D. and Sayama, H.: In preparation.

Using Dynamic Behavior Prediction to Guide an Evolutionary Search for Designing Two-Dimensional Cellular Automata

Gina Maira Barbosa de Oliveira[1] and Sandra Regina Cardoso Siqueira[2]

[1] Universidade Federal de Uberlândia, Faculdade de Computação (FACOM)
Av. João Naves D'Ávila, 2160 - Bloco B - Campus S. Mônica, 38400-902 Uberlândia, Brazil
[2] Universidade Presbiteriana Mackenzie, Faculdade de Computação e Informática (FCI)
Rua da Consolação 896, Consolação, 01302-907 São Paulo, Brazil
gina@facom.ufu.br, sandra@mackenzie.com.br

Abstract. The investigations carried out about the relationships between the generic dynamic behavior of cellular automata (CA) and their computational abilities have established a very active research area. Evolutionary methods have been used to look for CA with predefined computational abilities; one in particular that has been widely studied is the ability to solve the density classification task (DCT). The majority of these studies are focused on the one-dimensional CA. It has recently been shown that the use of a heuristic guided by parameters that estimate the dynamic behavior of 1D CA can improve the evolutionary search for DCT. The present work shows the application of three parameters previously published in the one-dimensional context generalized to the two-dimensional space: sensitivity, neighborhood dominance and activity propagation were used to evolve CA able to perform the two-dimensional version of the density classification task. The results obtained show that the parameters can effectively help a genetic algorithm in searching for 2D CA. A new rule was found which performed better than others previously published for the 2D DCT.

1 Introduction

One of the challenges to the artificial life field is to provide a theory explaining how dynamical systems can generate phenomena best understood as rule-based behavior [2]. In tune with that, this work is related to the idea of understanding the impact of the inherently local information processing of CA, and their ability to perform a coordinated computation at the global level, as mediated by an evolutionary process. Various investigations have been carried out on the computational power of CA, with concentrated efforts in the study of one-dimensional CA capable of performing computational tasks [1, 8, 12, 13, 15, 17, 18, 20, 23]. The most widely studied computational task is the density classification (DCT) [13], where the goal is to find a binary cellular automaton that can classify the density of 1s in the initial configuration of its lattice. One of the approaches in this kind of research is the use of Genetic Algorithms (GA) [7] as a search procedure to find CA with the predefined computational behavior. Various evolutionary techniques have been described in

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 491 – 499, 2005.

literature to find radius 3, two-state, one-dimensional CA with such abilities [1, 8, 12, 15, 17, 18, 20, 23].

Cellular Automata (CA) exhibit a variety of dynamic behaviors, although they are formed by simple basic components. It has already been proved that the decision problem associated with forecasting CA dynamics is undecidable [5]. A successful approach for analyzing the dynamics of a cellular automaton is the static analysis of its transition rule. Various studies on CA dynamics have been carried out, based upon parameterizations of their rule space, and using these parameters as indicators of the dynamical features in studies [3, 10, 16, 25]. The majority of these studies are focused only on the one-dimensional CA. It has recently been shown that the use of a heuristic guided by parameters that estimate the dynamic behavior of one-dimensional CA can improve the evolutionary search for the DCT and other computational tasks [15, 17, 18].

In the present work three generalizations of forecast parameters previously published and defined in the one-dimensional space were used. The parameters are called sensitivity, neighborhood dominance and activity propagation. They were generalized for two-dimensional binary CA using Moore neighborhood with an arbitrary radius r [19]. We present results of the application of these parameters in the evolution of 2D CA rules in the density classification task (DCT). A rule was found, which is better than those previously published and is presented here by the authors.

2 Cellular Automata Dynamics

Basically, a cellular automaton consists of the cellular space and the transition rule. Cellular space is a regular lattice of N cells, each one with an identical pattern of local connections to other cells, and subjected to some boundary conditions. These cells are arranged in a d-dimensional space and the most studied are the one-dimensional and the two-dimensional arrangements. The transition rule establishes how the states will change through time, based on the current states of each cell and their immediate neighbors. k denotes the number of possible states in a cell. For one-dimensional CA, the neighborhood size m is usually written as $m = 2r + 1$, where r is called the radius. In the case of binary states (i.e., two-state) CA, the transition rule is given by a rule table, which lists each possible neighborhood with its output bit, that is, the update value of the centre cell of the neighborhood. For two-dimensional CA, the two most-used connectivity schemes are known as von Neumann and Moore neighborhoods. A von Neumann neighborhood is formed by 5 cells; the central one and its four adjacent cells: in the east, in the west, in the north and in the south. A Moore neighborhood is formed by nine cells using the same five cells of the von Neumann neighborhood plus the diagonal cells, which are adjacent to the central one. The Moore neighborhood can be extended to larger sizes and the parameter r (*radius*) can also be used. The two-dimensional, $r = 1$ CA uses the simplest Moore neighborhood formed by 9 cells. In the present work the emphasis is on the $k = 2$, $r = 1$ 2D CA.

Wolfram proposed a qualitative classification of CA behavior in four dynamic classes, which is widely known [24]. Li and Packard (1990) later proposed a refinement to the Wolfram classification [11], which divides the rule space into six classes: null (or homogeneous fixed-point), (heterogeneous) fixed-point, two-cycle, periodic, complex and chaotic. The dynamics of a cellular automaton is associated

with its transition rule. Several parameters have been proposed directly calculated from CA transition rule in order to help forecast their dynamic behavior. The high cardinalities of CA rule spaces make their parameterization a difficult task and a single parameter is not sufficient to capture all singularities; hence, a set of parameters is required [3, 16]. The majority of the published parameterizations are focused on one-dimensional, two-state CA. Some of the published forecast parameters are: the precursor parameter proposed by Langton (1990) known as λ parameter [10]; sensitivity (μ) [3], Z parameter [25], absolute activity [16], neighborhood dominance [16] and activity propagation [16]. It has been proved that it is not possible to expect the precise forecasting of a generic cellular automaton, from any arbitrary initial configuration once the decision problem associated with the latter generic proposition is undecidable [5]. Hence, all we can expect is really a parameter that can help forecast the dynamic behavior of a cellular automaton.

A set of five forecast parameters was selected and applied to the evolution of 1D DCT rules in [15] and it was able to aid in the evolutionary search. Subsequently, it was possible to analyze that the more relevant parameters in this task were sensitivity (μ), neighborhood dominance (*nd*) and activity propagation (*ap*). So, these parameters were chosen for the generalization in the two-dimensional context. Their formal definitions are presented in [19]. The central idea in all forecast parameters is to carry out a simple calculus analyzing all the transitions of the neighborhoods in a rule. The subjacent concepts to the parameters definitions in the one-dimensional space were maintained in the proposed generalization. A 2D cellular automaton rule with the Moore neighborhood (9 cells) is formed by 512 (2^9) different neighborhoods in the form $\begin{bmatrix} s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 \\ s_7 & s_8 & s_9 \end{bmatrix}$. These neighborhoods can be linearly represented by $s_1\ s_2\ s_3$-$s_4\ s_5\ s_6$- $s_7\ s_8\ s_9$ and the output bits of the rule can be lexographically ordered from the neighborhood 000-000-000 to the 111-111-111.

Sensitivity (μ) is defined as the number of changes in the outputs of the transition rule, caused by changing the state of a cell of the neighborhood, one cell at a time, over all possible neighborhoods of the rule being considered. For example, for the neighborhood 000-000-000, nine flipped neighborhoods must be analyzed from 000-000-001 to 100-000-000. Let us suppose that the output bit related to neighborhood 000-000-000 is 1. In this case, all the flipped neighborhoods with the output equal to 0 will be considered sensitive. The parameter counts the number of sensitive flipped neighborhoods, over all possible neighborhoods of the rule being considered. The maximum count for this parameter in the case of 2D CA with Moore neighborhood is 4608. This value is used to normalize the parameter between 0 and 1.

Neighborhood dominance (*nd*) quantifies how much change is entailed by the CA rule transitions, in the state of the centre cell, in respect to the state that predominates in the neighborhood as a whole. For example, in the transition 0**1**0-**111**-**11**0 → **1**, neighborhood dominance occurs because the state that predominates in the neighborhood is "1" and the transition maps the centre cell state onto "1"; this is in contrast to transition 0**1**0-**111**-**11**0 → **0**, where dominance does not occur. The parameter value comes from a weighed sum of the number of transitions of the CA rule in which neighborhood dominance occurs, with the additional feature that, the

more homogeneous the neighborhood involved, the higher its weight. For example, the maximum weight 126 is given to the transition related to the totally homogeneous neighborhood 000-000-000, while the most heterogeneous neighborhoods, as 010-110-001 and 100-010-111, are given the minimum weight 1. The maximum count for this parameter is 6120 and it is also used in its normalization.

Activity Propagation (*ap*) is defined from two concepts related to the definitions of *nd* and μ: the possibility of a transition "following" (or not) the state that dominates the neighborhood, and the possibility of a transition being sensitive to a minimal change (a single state flip) in the neighborhood. First, each neighborhood in which the dominance does not occur is considered active. Subsequently, for each active neighborhood the calculation checks how many of the nine flipped neighborhoods related to the current one still remain active. For example, in the transition 000-000-000 → 1 dominance does not occur and it is considered active. Next, the flipped neighborhood 000-000-001 is considered. If the related transition is 000-000-001 → 1, this is also an active neighborhood and this occurrence counts as 1 in the parameter calculus. The other eight related neighborhoods are analyzed and each activity found counts as 1 in the parameter. This same analysis is made for each active neighborhood in the rule transition and its nine related flipped neighborhoods. The maximum value for this parameter is 4608 and it is used in its normalization.

3 Cellular Automata Computability and the Density Classification Task

The relationships between the dynamic behavior of a cellular automaton and its computational abilities have been studied as part of the more encompassing theme of the relationships between dynamical systems and computational theories [24]. The computational power of CA has been investigated with emphasis in the study of one-dimensional CA able to perform computational tasks [1, 8, 12, 13, 15, 20, 23]. The most widely studied CA task is known as the density classification task (DCT) [13]. In this task, the goal is to find a binary cellular automaton that can classify the density of 1s in the initial configuration of the lattice, such that: if it has more 1s than 0s, the automaton should converge to a null configuration composed of 1s; otherwise, it should converge to a null configuration of 0s. It was proven to be impossible to solve the DCT perfectly, by any one-dimensional cellular automaton with finite radius and periodic boundary conditions [9]. It is worth pointing out, though, that perfect solutions can be given to alternative formulations of the task, such as by allowing the cellular automaton to have non-periodic boundary conditions [22], by allowing the application of two distinct elementary CA rules in sequence [6], or by changing the classification criterion [4]. All in all, the best possible imperfect rule for the DCT - under the original formulation - remains unknown. The trend to look for better and better DCT rules has led to better and better algorithms, more and more fine-tuned to the problem [1, 8, 12, 13, 15, 20, 23].

Once a computational task is defined, it is far from trivial finding CA rules that perform it, due to the high cardinalities of the rule spaces. A practical alternative is the usage of evolutionary computation methods, as the genetic algorithms (GA) [7]. Packard (1988) was the first to publish results using a GA as a tool to find CA rules

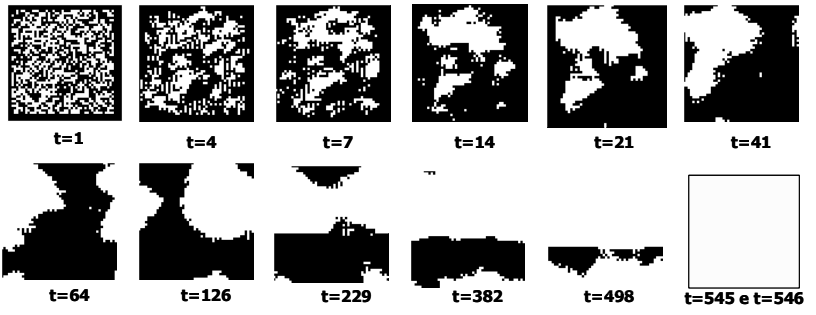


Fig. 1. Example of a CA performing 2D TCD (Moore neighborhood)

with a specific computational behavior [20]. He considered CA rules as individuals in a population and defined their fitness according to their ability to perform the specified task. Since Packard's work, several investigations have been carried out on using evolutionary methods to solve one-dimensional DCT [1, 8, 12, 13, 15, 23].

A forecast parameter set can be used as an auxiliary metric to guide the processes underlying the GA search. This approach has already been applied for the 1D DCT [15]. Once information was available on parameter value regions where good rules should be more likely to occur, it was used in genetic searches in which both the efficacy of the best rule found as the average efficacy have improved [15].

Morales *et al.* (2000) have extended the DCT for the two-dimensional space [14]. In this version, the goal is to find a 2D cellular automaton that can classify the density of 1s in the initial configuration of the 2D lattice. Figure 1 shows an example of a 2D CA rule performing the DCT task for an initial lattice with more 0's than 1's. Morales and his colleagues have used an approach similar to that used in 1D DCT and a GA has been used to find the 2D rules [14]. The best rule found has an efficacy of 69% in performing 2D DCT. Later, Reynaga and Amthauer (2003) extended Land and Belew's proofs for 1D DCT and they proved that it is also impossible to solve the 2D DCT perfectly, by any two-dimensional cellular automaton with finite radius [21]. Apparently without prior knowledge of Morales' work, they presented a 2D CA rule that was manually constructed inspired in GKL's one-dimensional rule [13]. They claimed that their rule has a reasonable performance in 2D DCT [21]. However, this rule has a poor performance in the most difficult cases of classification; initial lattices with approximate 50% of the bits in state 1 characterize these cases. So, several rules found by Morales and colleagues outperform that constructed by Reynaga and Amthauer.

4 Evolving Cellular Automata for 2D Density Classification Task

We first replicated Morales and colleagues' simulation, performing a series of 100 GA runs. We used the same configuration described in [14]: binary radius 1 Moore neighborhood CA were used, with 2D lattice formed by 21×21 cells. GA evolved a population of 100 rules during 100 generations. Each individual evaluation was obtained by testing the efficacy of the rule in 100 initial configurations (IC). Elitism

[7] was used at a rate of 20%, parent selection for the one-point crossover was made directly from the elite and mutation was applied after crossover at a rate of 2% per bit. The efficacy of the GA run was measured by testing the efficacy of the best rule found in the classification of 10^4 totally random ICs (the most difficult cases of classification). The efficacy average found in 100 GA runs was 56.58% and the efficacy of the best rule found was 67.57%.

The analysis of the best rule obtained in each run gave us the initial evidence that the generalization of the parameters was successfully realized. We simulated the 100 rules obtained after different initial configurations and observed the dynamic behavior performed by each one. Basically, we found null, fixed-point and two-cycle rules. However, two groups of rules were strongly characterized: the first one is formed by 22 two-cycle rules with efficacy between 43% and 55% and the second one is formed by 40 null rules with efficacy between 60% and 70%. We calculated the values of the generalized parameters for each rule in these groups. Table 1 shows the results obtained. By comparing the values between the two groups, one can see that for sensitivity, the null rules are concentrated in a narrow range that is inside the range found for two-cycle group. The two groups are concentrated in specific ranges for neighborhood dominance and activity propagation parameters. So, the dynamical behavior seems to be well characterized by parameters values. A comparative analysis was performed and it was possible to observe that the values obtained for the generalized parameters are coherent with the original one-dimensional versions. This analysis is presented in reference [19]. As it is already known what kind of behavior is desired, the null one, the bands related to this dynamics can be used as useful information to guide the genetic search.

Table 1. Parameter ranges found for 2D and for elementary rules

	μ	*nd*	*ap*
2D-radius 1 null rules	0.42 to 0.48	0.79 to 0.84	0.11 a 0.16
2D-radius 1 two cycle rules	0.25 to 0.48	0.22 to 0.53	0.25 to 0.38

Subsequently, simulations were performed where the selected parameter set was used as an auxiliary heuristic in evolutionary searching for CA. This approach is similar to the one used in one-dimensional experiments described in [15] and [17]: the parameter-based heuristic was coded as a function (referred to as *Hp*), which returns a value between 0 and 100 for each transition rule, depending on the rule's parameters. In the present experiments, the parameter ranges found for the 2D *radius* 1 null rules presented in Table 1 were used as the desirable ranges. Function *Hp* was defined so as to return 100 if all the parameters of a cellular automaton rule matched those of the desirable ranges; otherwise, the returned value would decrease as the parameter values moved increasingly away from those ranges. The GA was modified to incorporate the parameter-based heuristic in two aspects. First, the fitness function of a rule was made by the weighted average of the original fitness function (efficacy in 100 different ICs) and the function *Hp*. Second, biased crossover and mutation were used: in order to choose the crossover point and the bits to be mutated, various attempts were made and those that generated rules with high *Hp* were selected.

Seven parameter-based simulations were carried out, each one formed by 100 GA runs, using all possible combinations of the three parameters: individually, two-by-two and the three parameters simultaneously. In all the simulations it was possible to observe that the incorporation of the parameter-based heuristics returns better results from the evolutionary search, as in terms of efficacy average as in best efficacy found. The three best simulations were selected: the first uses sensitivity and activity propagation (efficacy average of 60.95%), the second uses neighborhood dominance and activity propagation (efficacy average of 65.03%) and the third uses the three parameters (efficacy average of 65.09%).

A second round of simulations were performed using a more elaborated genetic algorithm. In several situations where evolutionary techniques have been used the number of possible instances of the problem is high, disabling the exhaustive test of a solution. Therefore, sampling the instances is necessary to evaluate the solution population. An important issue for improving the statistical estimates in such cases is how to sample test cases and then weigh their contribution to fitness estimates. Several strategies have been proposed to aid this problem in order to preserve population diversity, such as shared sampling, competitive fitness functions, and resource sharing fitness functions [8, 23]. The basic idea underlying resource sharing is to give higher fitness to solutions that are able to solve test cases that are unsolvable by a large fraction of the solution population. From a different perspective, it can also be said that solutions receive smaller reward for pursuing strategies that put them into niches already heavily occupied. Resource sharing is intended to preserve diversity, so as to prevent mediocre solutions from taking over the population [23]. In the original work of Juillé and Pollack (1998) [8], where the best-known one-dimensional DCT rule was obtained, they used a combination of coevolutionary algorithm and resource sharing. Later, Werfel and collaborators (2000) investigated the relative contribution of each of the two techniques and concluded that the good results obtained by Juillé and Pollack were largely due to resource sharing [23].

A new parameter-based evolutionary environment was elaborated where the resource sharing was used in the individual evaluations, as described in references [8] and [23]. Four simulations were performed in this new environment and their results are presented in Table 2. In these simulations, the population size and the number of initial configurations used in each evaluation were increased to 160. We also raised the number of generations to 500. All the other GA and CA parameters were maintained from the initial simulations. Each simulation was formed by 20 GA runs. The first one was carried out without the parameter heuristic and we called it *RSGA(0)*. The other three were performed using the three best parameter combinations cited previously; we called them *RSGA(μ+ap), RSGA(nd+ap)* and *RSGA(μ+nd+ap).* Looking at the efficacy bands of the found rules, the parameter-based simulations have clearly shifted the bands to higher performance ones, when compared to the experiment *RSGA(0)*, performed with no heuristic. Table 2 also shows the average efficacy out of 20 runs as well the efficacy of the best rules found and making it evident that the rules found by using the parameter information have higher efficacies than those found with no information. The best rule was found in *RSGA(nd+ap)* which presents an efficacy of 70.84%. Later, this rule was submitted to more extensive tests and we can affirm that its performance is better than the best one found in Morales and colleagues' experiments [14].

Table 2. Efficacy bands, average and best rules found in 2D DCT simulations

Efficacy Bands	# of rules found in *RSGA(0)*	# of rules found in *RSGA(μ+ap)*	# of rules found in *RSGA(nd+ap)*	# of rules found in *RSGA(μ+nd+ap)*
[40%, 60%]	9	1	0	0
[60%, 65%]	5	3	8	8
[65%, 70%]	6	15	11	12
[70%, 75%]	0	1	1	0
Average	58.07%	65.59 %	65.54 %	65.83%
Best Rule	68.20%	70.01%	70.84%	68.87%

5 Conclusions

Three forecast parameters previously published in the one-dimensional context were generalized for the two-dimensional space and they were used to aid 2D DCT search. The results presented confirm that the use of the parameters information as a heuristic improved the underlying search performed by a genetic algorithm with a resource sharing strategy. A good rule for 2D density classification task was found. The hexadecimal code of the new rule is: 020D0311-0C191449-230F4B1D-85D35577-114900D5- 91AF5CB7-56151B1D-91FFF1FF-4320100B-2319357F-2F5C6777-8E5F5BBB-126E13F6-5639718F-076B7CF7-CED75777. As far as we know, it is the best rule found for this two-dimensional task and was found that the efficacy of this rule still remains very much above of the efficacy of good rules found for 1D DCT. The efficacy of the best rule found for one-dimensional DCT is 86% [8]. Based on this information, we speculate that there still exists a lot of room for improving the results in the two-dimensional version of DCT.

Finally, further investigations should be carried out to use the parameter-based heuristic in other computational tasks that can be generalized to 2D CA space, like the Synchronisation Task [18].

Acknowledgements

This work was partially supported by CNPq (PQ: 304639/2004-4), CAPES and MACKPESQUISA.

References

1. Andre, D., Bennett III, F., Koza, J. Discovery by Genetic Programming of a Cellular Automata Rule that is Better than any Known Rule for the Majority Classification Problem. In: Proceedings of Genetic Programming 96. Stanford: Stanford University (1996)
2. Bedau, M., McCaskill, J., Packard, N., Rasmussen, S., Adami, C., Green, D., Ikegami, T., Kaneko, K., Ray, T. Open Problems in Artificial Life. Artificial Life, 6(4) (2000) 363-376
3. Binder, P. A Phase Diagram for Elementary Cellular Automata. Complex Systems, 7 (1993) 241–247
4. Capcarrère, M., Sipper, M., Tomassini, M. Two-state, r=1, cellular automata that classifies density. Physical Review Letters, 77(24) (1996) 4969-4971

5. Culik II, K., Hurd, L., Yu, S. Computation Theoretic Aspects of Cellular Automata. Physica D, 45 (1990) 357-378.
6. Fukś, H. Solution of the density classification problem with two cellular automata rules. Physics Review E, 55 (1997) 2081R-2084R.
7. Goldberg, D. Genetic algorithm in search, optimization and machine learning. Addison-Wesley (1989)
8. Juillé, H., Pollack, J. Coevolving the "Ideal" Trainer: Application to the Discovery of Cellular Automata Rules. In: Proceedings of Genetic Programming Conference. Madison, 3 (1998)
9. Land, M., Belew, R. No Perfect Two-State Cellular Automata for Density Classification Exists. Physical Review Letters, 74(25) (1995) 5148-5150
10. Langton, C. Computation at the Edge of Chaos: Phase Transitions and Emergent Computation. Physica D, 42 (1990) 12-37
11. Li, W., Packard, N. The Structure of Elementary Cellular Automata Rule Space. Complex Systems, 4 (1990) 281-297
12. Mitchell, M., Hraber, P., Crutchfield, J. Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments. Physica D, 75 (1994) 361-391
13. Mitchell, M. Computation in Cellular Automata: A Selected Review. In: Nonstandard Computation. Weinheim: VCH Verlagsgesellschaft (1996)
14. Morales, F., Crutchfield, J., Mitchell, M. Evolving two-dimensional cellular automata to perform density classification: a report on work in progress. Parallel Computing, 27 (2000) 571-585
15. Oliveira, G., de Oliveira, P., Omar, N. "Evolving Solutions of the Density Classification Task in 1D Cellular Automata, Guided by Parameters that Estimate their Dynamic Behavior". In: M. A. Bedau, J. S. Mc Caskill, N. H. Packard and S. Rasmussen, (eds.): Proceeding of Artificial Life VII (2000) 428 – 436
16. Oliveira, G., de Oliveira, P., Omar, N. Definition and Applications of a Five-Parameter Characterization of One-Dimensional Cellular Automata Rule Space, Artificial Life, 7(3), MIT Press (2001) 277-301
17. Oliveira, G., de Oliveira, P., Omar, N. Searching for one-dimensional cellular automata in the absence of a priori information. In: J. Kelemen and P. Sosík, (eds.): Advances in Artificial Life (Proc. of the 6th European Conf. in Artificial Life, held in Prague, Czech Republic, Sept. 10-14), Lecture Notes in Artificial Intelligence 2159, Springer-Verlag, Berlin (2001) 262-271
18. Oliveira, G., de Oliveira, P., and Omar, N. Improving Genetic Search for One-Dimensional Cellular Automata, Using Heuristics Related to Their Dynamic Behavior Forecast. Proc. of the 2001 IEEE Conference on Evolutionary Computation, Seoul, South Korea, IEEE Press, Piscataway, NJ, USA (2001) 348-355
19. Oliveira, G., Siqueira, S. Parameter Characterization of Two-Dimensional Cellular Automata Rule Space. Submitted to Physica D (2005)
20. Packard, N. Adaptation toward the Edge of Chaos. In: Dynamic Patterns in Complex Systems. World Scientific, Singapore, (1988) 293-301
21. Reynaga, R., Amthauer, E. Two-dimensional cellular automata of radius one for density classification task ρ =1/2. Pattern Recognition Letters, 24 (2003) 2849–2856
22. Sipper, M., Capcarrère, M., Ronald, E. A simple cellular automaton that solves the density and ordering problems. International Journal of Modern Physics, 9(7) (1998) 899-902
23. Werfel, J., Mitchell, M., Crutchfield, J. Resource Sharing and Coevolution in Evolving Cellular Automata. IEEE Transactions on Evolutionary Computation, 4(4) (2000) 388-393
24. Wolfram, S. Computation Theory of Cellular Automata. Communication in Mathematical Physics, 96 (1984) 15-57
25. Wuensche, A. Classifying Cellular Automata Automatically: Finding gliders, filtering, and relating space-time patterns, attractor basins and the Z parameter. Complexity, 4 (3) (1999) 47-66

CelloS: A Multi-level Approach to Evolutionary Dynamics

Camille Stephan-Otto Attolini[1], Peter F. Stadler[1,2], and Christoph Flamm[1]

[1] Institut für Theoretische Chemie,
Universität Wien, Währingerstraße 17, A-1090 Wien, Austria
{camille, studla, xtof}@tbi.univie.ac.at
[2] Lehrstuhl für Bioinformatik, Institut für Informatik, Universität Leipzig,
Kreuzstraße 7b, D-04103 Leipzig, Germany
{camille, studla, xtof}@bioinf.uni-leipzig.de

Abstract. We study the evolution of simple cells equipped with a genome, a rudimentary gene regulation network at transcription level and two classes of functional genes: motion effectors which allow the cell to move in response to nutrient gradients and nutrient importers required to actually feed from the environment. The model is inspired by the protist *Naegleria gruberi* which can switch between a feeding and dividing amoeboid state and a mobile flagellate state depending on environmental conditions. Simulation results demonstrate how selection in a variable environment affects the gene number and efficiency making the cells to rapidly switch from one expression regime to the other depending on the external conditions.

Keywords: Artificial Cells, gene regulation, evolution, *Naegleria gruberi* .

1 Introduction

A non-trivial task in Artificial Life research is to devise genotype-phenotype maps, i.e., relations between genomic sequence information and the shape, structure and behavior of the organism encoded by the genome. The difficulties stem from the complexity of even the simplest cells, which precludes a representation of an entire cell at the molecular level. At present there are no established "intermediate-level" theories that would provide consistent but simplified representations of cellular processes (energy metabolism, biomass production, cell division, sensory responses, intracellular transport, gene expression, etc.). One therefore has to resort either to simulations based on a large number of *ad hoc* assumptions, or to the construction of minimal models based on biophysical and biochemical principles.

The process of RNA folding, for example, can be viewed as a minimal model of a genotype-phenotype map. Here, the sequence of the RNA molecule acts as the genotype (the sequence information is actually heritable in *in vitro* selection (SELEX) experiments [16]), while the (secondary) structure of the molecule is interpreted as the phenotype (SELEX experiments indeed often demonstrate a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 500–509, 2005.

strong structure dependence of the selected nucleic acids). Detailed investigations of the RNA model lead to the development of important concepts, such as neutral networks percolating sequence space, the phenomenon of shape space covering, and the importance of accessibility for phenotypic evolution [20,7]. The structure of the genotype-phenotype map determines the structure of the fitness landscape [21] which in turn determines the dynamics of an evolving population. The high degree of neutrality of the RNA folding map, for example, explains punctuated equilibria in the absence of external events [8,13], leads to a selection for robustness against mutations [22] and influences evolvability [4].

Concepts such as epistasis and phenotypic plasticity easily translates into this RNA folding metaphor [6], however, important characteristics of the genotype-phenotype maps of biological organisms, do not have a counterpart in this framework: While genotype and phenotpye are embodied in the same physical entity in the RNA model, there is a rather strict separation between genomic information and functional molecules in all biological organisms. This allows an organism to exist in different internal states (that depend on its *individual* history) which may cope with environmental conditions in different ways. Regulatory networks are at the core of the mechanism by which cells individually adapt to changing conditions, see e.g. [9,3]. The majority of the artificial gene regulation models used today [1,5,11,19] are based on the well established "operon model" of gene expression [14], which divides the genes into two classes: (i) the transcription factors capable of binding to the DNA and thereby modulating the expression of downstream located genes; and, (ii) structural proteins which perform some functions different from the regulation of the gene expression. In the simplest case, regulatory networks arise when transcription factors also enhance or inhibit the expression of other transcription factors. (Note that such models still ignore crucial regulation mechanisms of real cells such as signal transduction networks and post-transcriptional gene silencing.)

Our approach is motivated by the cell differentiation of the amoeba *Naegleria gruberi*, which is capable of changing cell shape, from a crawling amoeba to an asymmetric elongated cell, and of growing flagella when nutrients are scarce. It has been shown [10] that all proteins necessary for the differentiation are synthesized *de novo*, i.e., due to transcriptional regulation. The initiation of morphological changes require the synthesis of sufficient amounts of proteins, i.e., a significant investment. The transformation is temporal and the organism returns back to the amoeba state when nutrients are again available.

The `CelloS` model described in this contribution combines a simple computational cell model which includes a notion of space, the extended Potts model (see [18] and references therein), with an artificial genome and a minimal model of gene expression [19]. Given the high complexity of the genotype-phenotype map, we chose to keep the various modules of the model as simple as possible in order to understand how the composite system generates its intricate behavior. Regardless of the computational constraints, too detailed models are difficult to analyze and main features may be lost while working with large amounts of data. This combination of the parts allows to study the coupling of the environmental

dynamics to the internal dynamics of gene regulation within the framework of an evolving cell population.

In the next section the model is described, followed in section 3 by some results obtained so far in this project. Section 4 presents future goals of `CelloS` and some topics to be addressed with more complex implementations of the model.

2 The Model

The basic tool for our simulations is the Potts model with some extensions (for a complete description of the model refer to [17]) on a 2D lattice. A *cell* C is a maximal connected subset of the lattice such that all lattice points in C have the same type or "color" u. Cells interact with each other with strength J_{uv} at neighboring lattice points depending on their types u and v. A special type 0 denotes empty lattice sites. Each cell is characterized by its energy

$$E_C = \sum_{i \in \partial C} \sum_{j \in N(i) \setminus C} J_{u_C, u_j} + \lambda(\mathrm{vol}(C) - V)^2 \tag{1}$$

where $\mathrm{vol}(C)$ is the volume of the cell, i.e. its number of lattices points, ∂C its boundary, V is a user-defined target volume, $N(i)$ the set of neighbors of i and λ a compressibility parameter. The double sum runs over all boundary (surface) points of the cell C which are in contact with other cells or with the environment. A certain number of nutrient sources are randomly distributed in the environment. Nutrient concentration c_i at lattice point i is maximal in sources and decreases proportionally to distance.

Cell motion is implemented by a simple Metropolis Monte Carlo step in which a cell attemps to modify its boundary at lattice point $i \in \partial C$ by changing the type of an adjacent site i' to its own type, or by changing one of its boundary sites to 0. The cells feel the gradient in the nutrient by evaluating $c_i - c_i$. The transition probability is $\min\{1, \exp(-(\Delta E_C + \mu_0(c_i - ci) + H_\partial)/T)\}$ where H_∂ is the energy cost of deforming the cell's boundary, μ_0 describes the reactivity of the cell to changes in the nutrient concentration, and T a temperature-like parameter representing the default motility of the cells. Our cells have a finite life expectancy and require energy to stay alive. This is modeled by a "battery" which is used up when enzymes are synthesized. When the "battery" is empty, the cell dies and the corresponding lattice sites are reset to 0.

Each cell on the lattice contains an RNA sequence of length 1000 which represents its genome and carries the information needed to decode the cell's behavior. This genome can encode two types of effector molecules (corresponding of course to proteins in *N. gruberi*, but modeled as RNAs here for computational convenience) and a simple regulation mechanism. The "genetics" of the `CelloS` model is summarized in Fig. 1.

A short signal sequence (corresponding e.g. to the TATA box in real cells) marks the beginning of a "coding region" on the genomic sequence. We use the signal GC and define a gene to be the following 40 nucleotides. This subsequence

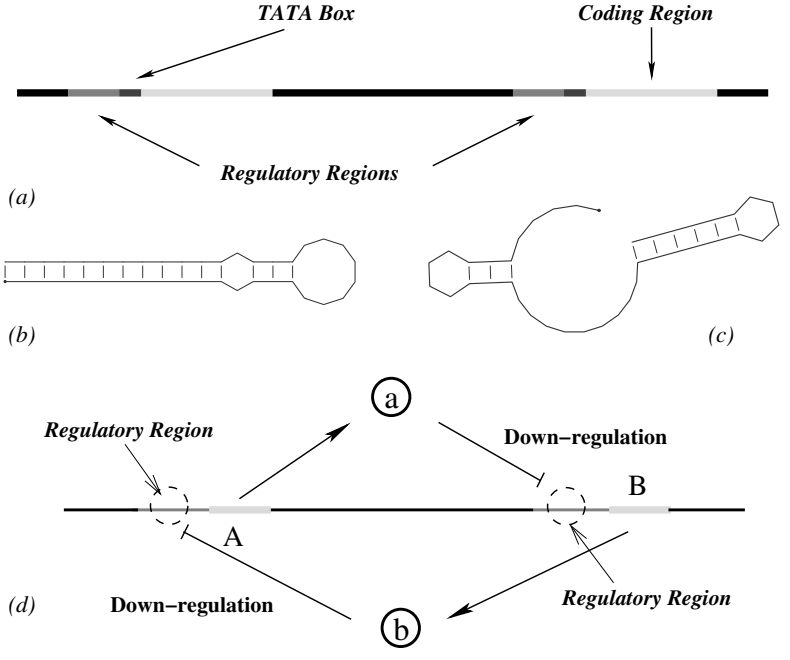


Fig. 1. Genetics of the `Cellos` model. (a) Genomic organization. Two classes of functionally different RNAs are distiguished by archetypic shapes: motion effectors (b) and metabolic effectors that act as nutrient importers (c). Panel (d) summarizes the logic of regulation in `Cellos`: expression is down-regulated when an RNA from the other function-class binds to the regulatory region of the gene.

is folded into its secondary structure using the `RNAfold` program of the `Vienna RNA Package` [12]. This structure is then compared with two target shapes for the "motion effectors" and the "nutrient importers", which are kept fixed throughout the simulation. The closer target shape determines the function of the gene, while the number of base pairing differences measures the gene's efficiency.

In the current implementation we keep the gene regulation network fixed. In order to implement the switching between the motion effectors and nutrient importers we use the simple negative feedback system shown in Fig. 1 (bottom). The differential equations for this scheme are:

$$\begin{aligned} \frac{dG_A}{dt} &= \gamma_A \cdot k \frac{1}{1+G_B^3} - d \cdot G_A \\ \frac{dG_B}{dt} &= \gamma_B \cdot k \frac{1}{1+G_A^3} - d \cdot G_B \end{aligned} \tag{2}$$

where G_A and G_B are the concentrations of the two types of gene products, γ_A and γ_B are their efficiencies, and k and d fixed constants. A 4th order Runge-Kutta method is used to numerically integrate these differential equations.

Information from the environment is given to the cell via impulses in the effectors concentrations. Whenever a cell is touching a food source, the concentration of nutrient importers is increased and the equations integrated. The same is done with motion effectors when no food is available. The battery level B is decreased depending on the effectors that are produced and it is recharged if the cell is in a food source:

$$B' = B - \psi_0(G_A + G_B) + \phi_0 G_B \tag{3}$$

The parameteres ψ_0 and ϕ_0 describe the ratio of nutrients obtained from the environment against the cost of producing the importers and motion effectors. The mobility of the cell depends on the concentration of expressed motion effectors which is reflected in a modified transition probability for changing the cells boundary by replacing the constant μ_0 with $\mu_0 \cdot G_B$.

The products of metabolic genes play two different roles: recharging the battery and increasing the cell's target volume. Once a cell has doubled its normal size, it divides by fission copying its genome to the new cell. This process is usually inaccurate, producing a point mutation in the new RNA string.

Food sources are depleted when cells feed from them. Once a source is empty, it is relocated in a randomly chosen spot of the lattice. This way, cells are forced to switch between the metabolic and movement states, reinforcing the selection of only those capable of doing so.

Individual cells with very similar genomes belong to the same *species*. The definition of species in our model is similar to that proposed by Kenneth and Risto in [15]. Each gene in the population has a unique historical number. Every time a mutation creates a new one or changes the type of an old one, this global variable is increased and assigned to the new gene. In order to compare two genomes, we use a linear combination of the number of excess (X) and disjoint (D) genes, and the average efficiency difference between common genes (W). Every time a new cell is born, its genome is compared to all species' genomes. If the result of

$$\delta = \frac{c_1 X}{N} + \frac{c_2 D}{N} + c_3 \cdot W \tag{4}$$

is below a threshold value, the cell is said to belong to the corresponding species. Otherwise its genome is set to represent a new species.

3 Simulation Results

Throughout all of our simulations we use a lattice of 200×200 sites with periodic boundary conditions. $J_{x,0} = 11$ for the contact with an empty site, $J_{x,y} = 37.5$ for the contact between different cell types, and $J_{x,x} = 35$ for the contact with a cell of the same species. These values reflect uniformity between the cells in their interaction with the environment. The lower value of $J_{x,x}$ compared with $J_{x,y}$ makes the accumulation of same-species cells a little more probable than between different species. Furthermore $T = 3$, $H_\partial = 0.8$, $\mu_0 = 5000$ are in a range were cells move fast enough to travel between sources in contrast with

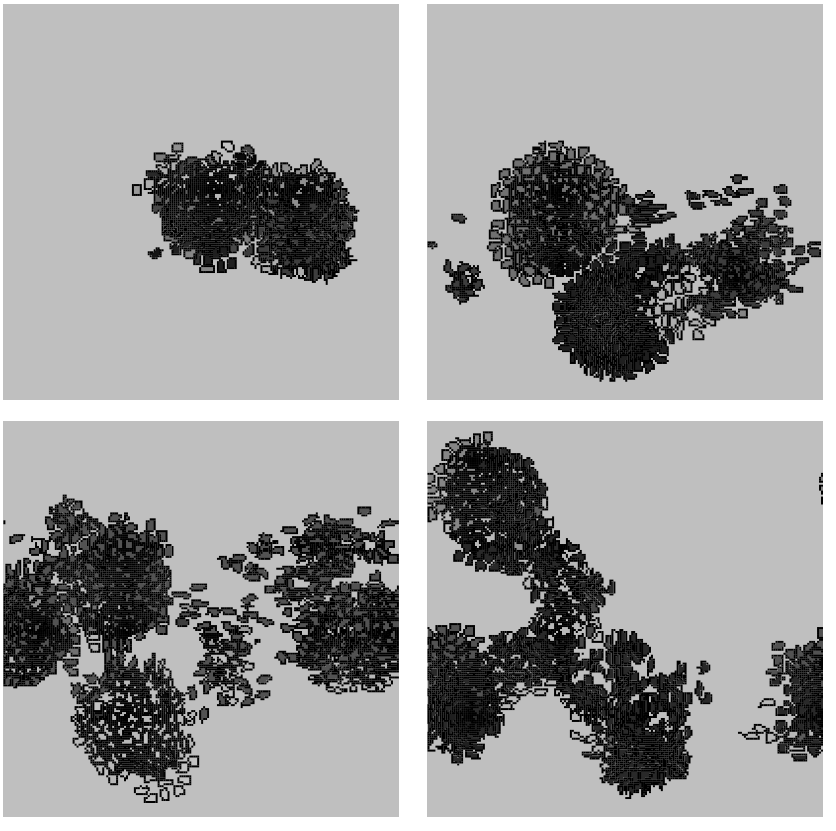


Fig. 2. Snapshots from a simulation run with three food sources at generations 135, 495, 12225 and 19800

their available energy and life span. $\psi_0 = 0.4$ and $\phi_0 = 2$ make the refilling of the battery to be enough compared with the sources life time. Other parameters are $V = 30$ and $\lambda = 5$.

Figure 2 shows the evolution of the system for one simulation run. In these images three food sources were available for the cells to eat. Positions of the sources are not shown but can be implied from the accumulation of cells in certain places. Population size changes depending on the conditions and parameters. A larger number of food sources is reflected in the increase of the population size.

3.1 Genome and Population Structure

We measure the impact of the external conditions in the genome by looking at the number of metabolic and movement genes, their efficiencies and the effectors expression inside and outside a food source.

The regulatory network we are using, imposes a well defined range in which gene efficiency must lay in order to obtain the necessary switch between states. In

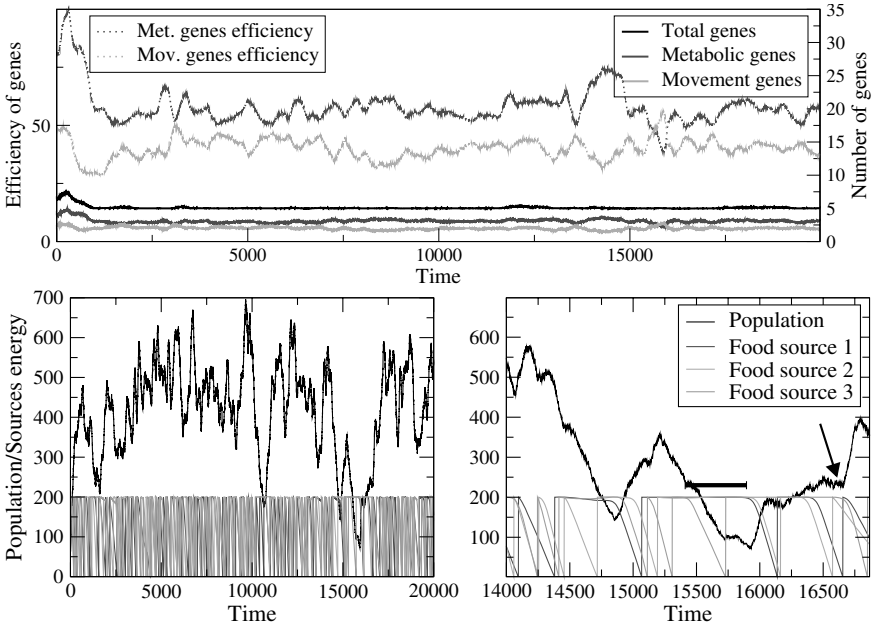


Fig. 3. Top: Number and efficiency of genes in a simulation with 3 food spots, mean life 600 and volume increment per generation 0.6. Bottom left: Population and energy of the sources. Bottom right: In the region under the bar the population is feeding from food source 2 only. At the right end of the bar, food source 3 is conquered. The arrow indicates the point where population feeds from all three sources and therefore increases dramatically.

our simulations it is clear how these numbers are controlled by natural selection when the genome is mutating randomly. In Fig. 3 it can be seen how after a period of adjustment, the population falls in a regime where gene number and efficiency are inside a small interval for both kind of genes.

The population grows depending on the availability of nutrients. Every time a food source is depleted, cells must migrate to the next one. These periods are usually reflected in a diminution of the population and increase in the average number of movement genes in it. The bottom figures in Fig. 3 show the energy of the sources and the change in the number of cells. Source energy staying at its maximum means there are no cells feeding from it, resulting in the decrease of the population size. Population can only increase or mantain its size when feeding from more than one food source (Fig. 3 bottom right).

This combination of gene types allows a switching in their products expression depending only in the presence or absence of food in the environment. Figure 4 shows this behavior for a single cell with the right number of genes.

As a special case we study the system with only one food spot of infinite life in the lattice. Cells in the spot are thrown out of it by the newborns. Even when there is no need of traveling long distances, the fact that cells have to be

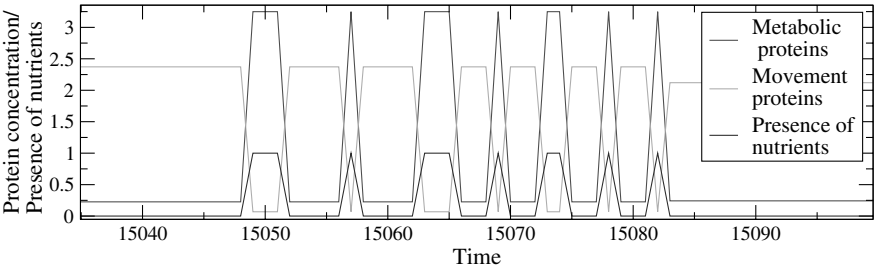


Fig. 4. Switching of gene products expression depending on the presence/absence of nutrients. Metabolic and movement genes efficiencies are 70 and 40 respectively. Same parameters as in the previous figure.

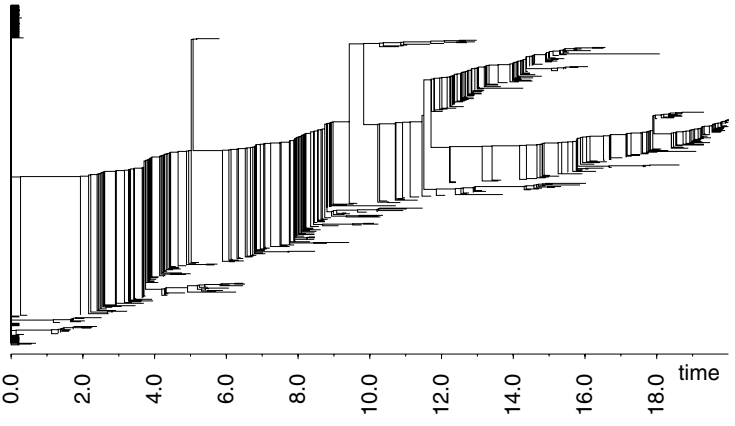


Fig. 5. Phylogenetic tree for a run with three food spots. Nodes in the tree represent the dissapearance of a species, while saddles stand for the split of two of them. Time unit is 1000 simulation steps.

constantly coming back into the source makes the presence of movement genes indispensable. At the same time, since food is easily available, there is no need to increase the efficiency of metabolic genes. Battery may be refilled slowly without killing the cell since the time it spends outside the food source is usually very short.

3.2 Phylogenetics

With our simple definition the number of species depends directly on the volume increase per generation. Phylogenetic trees can be recorded based on the speciation events, see Fig. 5 for a characteristic example. The Darwinian evolution is dominated by one or a few species at any given point in time. The coexistence of distinct lineages over longer times is comparably rare. These periods of coexistence are interrupted by points in time which act like bottlenecks where only

one species survives. In some runs one of the initial species survives until the end, failing to find any important improvement in phenotype via mutations.

4 Concluding Remarks

In this first (and very simple) implementation of the model, we observe the response of the genome to variable environmental conditions. After an initial phase of selection the number of genes stays approximately constant. The cells can then use their gene regulation network to cope with environmental changes. Population dynamics also reflect the presence or absence of nutrients, together with an increase of the number and/or efficiency of movement genes. We found that, at least in our simple environment, it is not important to have a large number of genes, but to have the right amount of them depending on the environmental inputs and the regulatory network modifying their products' expression.

Since the mechanism of regulating gene expression in the current implementation of the `CelloS` model can itself not be a target of evolution, we plan to add transcription factors as a third class of gene products to the artificial genome. This will allow the cells to find innovative regulatory strategies based on post transcriptional interaction. A fruitfull route will then be to study the mixing of regulatory strategies under sexuall reproduction of the cells.

Extending the set of mutation operators from point mutation to gene duplication and horizontal gene transfer, turns `Cellos` into a tool for generating test data for phylogenetic reconstruction methods. Comparing the simulated evolutionary scenario with the reconstructed one will allow to evaluate the performance of such methods.

The environmental dynamics can also be improved by switching to an artifical chemistry like the Toy Chemistry Model [2]. This forces for an additional decoding layer in the internal structure of the cells, which links our representation of the nutrient importers to organic molecules in the environment. Improvements of the `CelloS` model along these lines are under way.

Acknowledgments

This work was supported in part by Consejo Nacional de Ciencia y Tecnología, CONACyT, the DFG Bioinformatics Initiative (BIZ-6/1-2), and COST Action D27. We thank the Aegean Sea for its stimulating effect on this work.

References

1. Wolfgang Banzhaf. On the dynamics of an artificial regulatory network. In *Proc. ECAL03*, volume 2801 of *LNCS*, pages 217–227, Heidelberg, Germany, 2003. Springer-Verlag.
2. Gil Benkö, Christoph Flamm, and Peter F. Stadler. A graph-based toy model of chemistry. *J. Chem. Inf. Comput. Sci.*, 43:1085–1093, 2003.

3. A. Deckard and H. M. Sauro. Preliminary studies on the in silico evolution of biochemical networks. *ChemBioChem*, 5:1423–1431, 2004.
4. Marc Ebner, Mark Shackleton, and Rob Shipman. How neutral networks influence evolvability. *Complex.*, 7(2):19–33, 2001.
5. Peter Eggenberg. Evolving morphologies of simulated 3D organisms based on differential gene expression. In *Proc. ECAL97*, pages 205–213. The MIT Press/Bradford Books, 1997.
6. Walter Fontana. Modelling 'evo-devo'with RNA. *BioEssays*, 24:1164–1177, 2002.
7. Walter Fontana and Peter Schuster. Shaping space: The possible and the attainable in RNA genotype-phenotype mapping. *J. Theor. Biol.*, 194:491–515, 1998.
8. Christian V. Forst, Christian M. Reidys, and Jacqueline Weber. Evolutionary dynamics and optimization: Neutral Networks as model-landscape for RNA secondary-structure folding-landscapes. In *Proc. ECAL95*, volume 929 of *LNAI*, pages 128–147. Springer, 1995.
9. Paul François and Vincent Hakim. Design of genetic networks with specified functions by evolution *in silico*. *Proc. Natl. Acad. Sci. USA*, 101(2):580–585, 2004.
10. Chandler Fulton and Charles Walsh. Cell differentiation and flagellar elongation in *Naegleria gruberi*. *J. Cell Biol.*, 85:346–360, 1980.
11. Nicholas Geard and Janet Wiles. Structure and dynamics of a gene network model. In *Proc. CEC-2003*, pages 199–206. IEEE Press, 2003.
12. Ivo L. Hofacker. Vienna RNA secondary structure server. *Nucl. Acids Res.*, 31:3429–3431, 2003.
13. Martijn A. Huynen, Peter F. Stadler, and Walter Fontana. Smoothness within ruggedness: the role of neutrality in adaptation. *Proc. Natl. Acad. Sci. (USA)*, 93:397–401, 1996.
14. F. Jacob and J. Monod. On the regulation of gene activity. *Cold Spring Harbor Symp. Quant. Biol.*, 26:193–211, 1961.
15. Stanley O. Kenneth and Miikkulainen Risto. Efficient Reinforcement Learning through Evolving Neural Network Topologies. In *Proc. GECCO-2002*, pages 569–577, San Fransisco, 2002. Morgen Kaufman.
16. S. Klug and M. Famulok. All you wanted to know about SELEX. *Mol. Biol. Reports*, 20:97–107, 1994.
17. Athanasius F.M. Marée and Paulien Hogeweg. Modelling *Dictyostelium discoideum* Morphogenesis: the Culmination. *Bull. Math. Biol*, 64:327–353, 2002.
18. Roeland M. H. Merks and James A. Glazier. A cell-centered approach to developmental biology. *Physica A*, 2005. in press.
19. Torsten Reil. Dynamics of gene expression in an artificial genome – inplications for biological and artificial ontogeny. In *Proc. ECAL99*, volume 1674 of *LNCS*, pages 457–466, Berlin, 1999. Springer-Verlag.
20. Peter Schuster, Walter Fontana, P. F. Stadler, and I. L. Hofacker. From sequences to shapes and back: A case study in RNA secondary structures. *Proc. Roy. Soc. Lond.*, B225:279–284, 1994.
21. Peter F. Stadler. Fitness landscapes arising from the sequence-structure maps of biopolymers. *J. Mol. Struct. (THEOCHEM)*, 463:7–19, 1999. Santa Fe Institute Preprint 97-11-082.
22. Erik van Nimwegen, James P. Crutchfield, and Martijn A. Huynen. Neutral evolution of mutational robustness. *Proc. Natl. Acad. Sci. USA*, 96:9716–9720, 1999.

A Cytokine Formal Immune Network

Alexander O. Tarakanov[1], Larisa B. Goncharova[2], and Oleg A. Tarakanov[3]

[1] St. Petersburg Institute for Informatics, Russian Academy of Sciences,
14th line 39, St. Petersburg, 199178, Russia
tar@iias.spb.su
[2] Institute Pasteur of St. Petersburg,
Mira 14, St. Petersburg, 197101, Russia
goncharova_lara@mail.ru
[3] Department of Mathematics, St. Petersburg State University,
Universitetskaya nab. 7/9, St. Petersburg, 199034, Russia
light_ln2@hotmail.com

Abstract. This paper develops a mathematical model of immune network controlled by cytokines. A software implementation of the model has been applied to intrusion detection in computer network. The obtained results suggest that the performance of the model is unachievable for another approaches of computational intelligence.

1 Introduction

Cytokines (messenger proteins) are a group of biologically active mediator molecules that provide the intercellular interactions within the immune system. They are the central regulators of leukocyte growth and differentiation, being produced by a wide variety of cell types, targeting various cell subsets and exhibiting numerous biological activities.

Up to now more than 100 different human cytokines are identified. An increasing volume of experimental data suggests that cytokines play one of the central roles in the immune regulation as well as in the neuro-immune-endocrine modulation [1], [21]. Such concept of cytokines as a network modulating and switching several cascades of immune reactions [20] adjoins with the concept considering such molecules as a field or a milieu, which local properties mediate immune response [18].

There exists a relationship between cytokine levels in human body fluids and disease pathogenesis, including the inflammation and even depression [3]. Many types of cancers have taken advantage of the regulatory role of cytokines to down-regulate appropriate immune responses targeted at destroying cancer cells. They do this by secreting immunosuppressive cytokines that induce generalized and specific inhibition of immune responses [19]. So, the use of immunostimulatory cytokines as tumor vaccines has become a promising strategy in cancer immunotherapy [12].

Recent developments show that cytokines induce apoptosis (programmed cell death) in cancer cells [15], [27], [28]. The induction of apoptosis is associated with a dose-dependent inhibition of cancer cell division, and this activity has been demonstrated for a wide range of cancer types including bladder, breast, leukemia, melanoma, ovarian and prostate.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 510 – 519, 2005.

Apoptosis is a natural mechanism by which cells "commit suicide" when they have outlived their purpose, become defective, or have aged. Apoptosis prevents cells from accumulating and forming tumors. Understanding of the control of apoptosis in normal and malignant cells will help to improve the diagnosis and treatment of malignancies. The goal of many treatments, including chemotherapies is to induce malignant cells to undergo apoptosis. Current data also suggests that a cytokine may function as a dual-acting cytokine in which its normal physiological functions may be related to specific aspects of the immune system and over-expression culminates in cancer-specific apoptosis [8].

On the other hand, immunological approach [4], [16], [17] looks rather constructive as a basis for a new kind of computing [5], [6], [24], including its applications to fault detection in air- and spacecrafts as well as simulating of the natural immune system [7], [10], [11], [14], [22]. In such background, this paper develops a rigorous mathematical model of immune network with the cytokine controlled apoptosis and immunization. A software implementation of the model has been applied to the task of intrusion detection in a local area network (LAN) and tested on data of the UCI KDD archive [2]. The obtained results suggest that training time and accuracy of the model are beyond the possibilities of artificial neural networks and genetic algorithms [25], [26].

2 Mathematical Model

2.1 Cytokine Formal Immune Network

Definition 1. Cell is a pair $V = (c, P)$, where "cytokine" c is natural number $c \in N$, whereas $P = (p_1,..., p_q)$ is a point of q-dimensional Euclidian space: $P \in R^q$, and P lies within unit cube: $\max\{| p_1 |,...,| p_q |\} \le 1$.

Let distance ("affinity") $d_{ij} = d(V_i, V_j)$ between cells V_i and V_j be as follows:

$$d_{ij} = \max\left\{\left|(p_1)_i - (p_1)_j\right|,...,\left|(p_q)_i - (p_q)_j\right|\right\}. \quad (1)$$

Fix some finite non-empty set of cells ("innate immunity") $W_0 = (V_1,...,V_m)$ with non-zero distance between cells: $d_{ij} \neq 0$, $\forall i, j : i \neq j$.

Definition 2. Cytokine formal immune network (cFIN) is a set of cells: $W \subseteq W_0$.

Definition 3. Cell V_i recognizes cell V_k if the following conditions are satisfied: $c_i = c_k$, $d_{ik} < h$, $d_{ik} < d_{ij}$, $\forall V_j \in W$, $j \neq i$, $k \neq j$, where $h \ge 0$ is given "threshold of affinity".

Let us define the behavior ("maturation") of cFIN by the following two rules.

Rule 1 (Apoptosis). If cell $V_i \in W$ recognizes cell $V_k \in W$ then remove V_i from cFIN.

Rule 2 (Auto-Immunization). If cell $V_k \in W$ is nearest to cell $V_i \in W_0 \setminus W$ among all cells of cFIN: $d_{ik} < d_{ij}$, $\forall V_j \in W$, whereas $c_i \neq c_k$, then add V_i to cFIN.

Let W_A be cFIN as a consequent of application of apoptosis to all cells of W_0. Let W_I be cFIN as a consequence of auto-immunization of all cells of W_A by all cells of W_0. Note that the resulting sets W_A and W_I depend on the ordering of cells in W_0. Further it will be assumed that the ordering is given.

2.2 Mathematical Properties of cFIN

It is obvious that neither the result of apoptosis W_A nor the result of auto-immunization W_I can overcome W_0 for any innate immunity: $W_A \subseteq W_0$, $W_I \subseteq W_0$, $\forall W_0$. Consider more important and less evident properties of cFIN.

Proposition 1. For any innate immunity W_0 there exists threshold of affinity h_0 such that apoptosis does not change W_0 for any h less than h_0: $W_A = W_0, \forall h < h_0$.

Let h_0 be minimal distance (1) for any pair of cells of cFIN with the same cytokines:

$$h_0 = \min_{i,j}\{d_{ij}\}: c_i = c_j, i \neq j.$$

Then, according to Definition 3, none of the cells of cFIN can recognize other cells, because $d_{ij} > h_0$ for any pair of cells V_i and V_j. According to Rule 1, none of the cells can be removed from cFIN for any h less than h_0, because $d_{ij} > h$, $\forall h < h_0$, $\forall V_i, V_j \in W_0$. Thus, $W_A = W_0, \forall h < h_0$.

Proposition 2. For any innate immunity W_0 there exists threshold of affinity h_1 such that consequence of apoptosis and auto-immunization $W_1 = W_I(h_1)$ provides the minimal number of cells $|W_1|$ for given W_0 and any h: $|W_1| \leq |W_I(h)|$, $\forall h$, $\forall W_I \subseteq W_0$.

Let h_1 be maximal distance (1) for any pair of cells of cFIN with the same cytokines:

$$h_1 = \max_{i,j}\{d_{ij}\}: c_i = c_j, i \neq j.$$

Then, according to Definition 3, any cell V_i can recognize the nearest cell V_j if the last one has the same cytokine: $c_i = c_j$. Let W_- be the set of all such cells V_i. Then, according to Rule 1, $|W_A(h_1)| = |W_0| - |W_-|$, and such number of cells after apoptosis is minimal among any h: $|W_A(h_1)| \leq |W_A(h)|, \forall h$. Let W_+ be set of cells, which is added to $W_A(h_1)$ as a consequence of auto-immunization: $W_1 = W_A(h_1) \cup W_+$. It is also evident that W_+ is a subset of W_-: $W_+ \subseteq W_-$, and $|W_+|$ represents a number of "mistakes" of apoptosis when cFIN "kills" some cells,

which lead to further recognition errors. Such cells are then "restored" by auto-immunization (Rule 2). Let $W_* = W_- \setminus W_+$ be cells which yield apoptosis without further recognition errors. Then $|W_+| = |W_-| - |W_*|$. On the other hand: $|W_1| = |W_A(h_1)| + |W_+|$. Substitutions of $|W_A(h_1)|$ and $|W_+|$ lead to the following result: $|W_1| = |W_0| - |W_*|$. Thus, $|W_1| \le |W_I(h)|$, which proves Proposition 2.

2.3 Application of cFIN to Pattern Recognition

Let "epitope" ("antigenic determinant") be any point $P = (p_1,..., p_q)$ of q-dimensional Euclidian space: $P \in R^q$. Note that any cell of cFIN also contains an epitope, according to Definition 1.

Definition 4. Cell V_i recognizes epitope P by assigning him class c_i if the distance $d(V_i, P)$ between the cell and the epitope is minimal among all cells of cFIN: $d(V_i, P) = \min\{d(V_j, P)\}, \forall V_j \in W$.

Let pattern be any n-dimensional column-vector $Z = [z_1,..., z_n]'$, where $z_1,..., z_n$ are real values and (') is symbol of matrix transposing. Let pattern recognition be mapping of the pattern to an epitope: $Z \to P \in R^q$, and recognition of the epitope by the class of the nearest cell of cFIN. Let $A_1,..., A_m$ be n-dimensional training patterns with known classes $c_1,..., c_m$. Let $A = [A_1,..., A_m]'$ be training matrix of dimension $m \times n$. Consider singular value decomposition (SVD: see, e.g., [13]) of this matrix:

$$A = s_1 Y_1 X_1^{'} + s_2 Y_2 X_2^{'} + s_3 Y_3 X_3^{'} + ... + s_r Y_r X_r^{'},$$

where r is the rank of matrix A, s_k are singular values and Y_k, X_k are left and right singular vectors with the following properties: $Y_k^{'} Y_k = 1$, $X_k^{'} X_k = 1$, $Y_k^{'} Y_i = 0$, $X_k^{'} X_i = 0$, $i \ne k$, $k = 1,..., r$, $s_{k-1} \ge s_k$, $k > 1$.

Consider the following mapping of any n-dimensional pattern Z to epitope P:

$$p_k = \frac{1}{s_k} Z' X_k, \; k = 1,..., q, \; q \le r. \tag{2}$$

Note that formulas (2) can be treated as "binding energies" between "formal proteins" Z ("antigens") and X_k ("antibodies"), according to [24]. Note also, that any epitope obtained by application of formulas (2) to any training pattern lies within unit cube (see Definition 1), according to the above properties of singular vectors.

3 Software Implementation of cFIN

3.1 Pattern Recognition Algorithm

Based on the above mathematical model of cFIN, consider a description (in a pseudocode) of a pattern recognition algorithm:

```
Training
{
    1st stage training // map data to cFIN ("antigen processing")
    {
        Get training patterns;
        Form training matrix;
        Compute SVD of the training matrix;
        Store q singular values // "binding energies"
        Store q right singular vectors; // "antibody-probes"
        Store left singular vectors; // cells of cFIN
    }
    2nd stage training // compress data by cFIN's "maturation"
    { // compute consecutively for all cells of cFIN:
        Apoptosis;
        Auto-Immunization;
    }
}
Recognition
{
    Get pattern; // "antigen"
    Map the pattern to cFIN; // by formulas (2)
    Find nearest cell of cFIN;
    Assign class of the nearest cell to the pattern;
}
```

This algorithm has been implemented in a version of the immunochip emulator [22] using Visual C++ with build in assembler code of the cytokine affinity function (1) for three-dimensional (3D) Euclidian space ($q = 3$) and OpenGL tools for 3D visualization. Screenshot of the emulator is shown in Fig. 1.

3.2 Test Results

The known UCI KDD archive [2] has been used for testing the emulator. Lincoln Labs set up an environment to acquire nine weeks of raw TCP (transmission control protocol) dump data simulating a typical US Air Force LAN. They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks.

The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records.

A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP (internet protocol) address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes.

Two data files from UCI KDD archive has been used to test the emulator:

- File 1: kddcup_data_10_percent_gz.htm (7.7 MB);
- File 2: kddcup_newtestdata_10_percent_unlabeled_gz.htm (44 MB).

File 1 is the training data file. It contains 51608 network connection records. Any record (file string) has the following format, where parameters 2, 3, 4, 42 are symbolic, while other 38 parameters are numerical (real values):

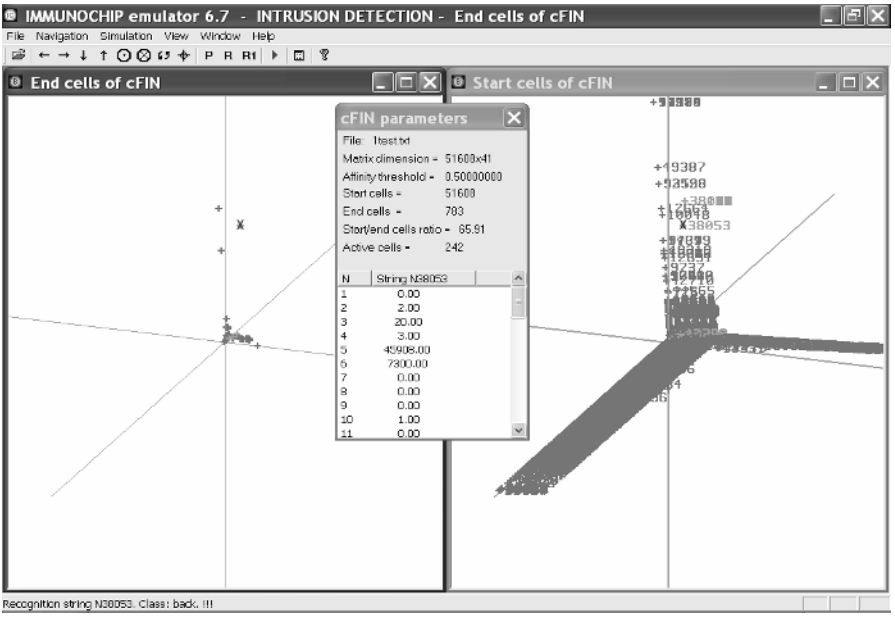


Fig. 1. Intrusion detection by cFIN: "Antigen" (String 38053 of File 1.1) is mapped to cFIN (bold skew cross) and recognized by the "cytokine" of the nearest cell of cFIN (Class: back !!!). Note also "tumors" (Start cells of cFIN in right-hand screen) eliminated after apoptosis and auto-immunization (End cells of cFIN in left-hand screen).

```
1) duration, 2) protocol_type, 3) service, 4) flag, 5) src_bytes,
6) dst_bytes, 7) land, 8) wrong_fragment, 9) urgent, 10) hot,
11) num_failed_logins, 12) logged_in, 13) num_compromised,
14) root_shell, 15) su_attempted, 16) num_root, 17) num_file_creations,
18) num_shells, 19) num_access_files, 20) num_outbound_cmds,
21) is_host_login, 22) is_guest_login, 23) count, 24) srv_count,
25) serror_rate, 26) srv_serror_rate, 27) rerror_rate,
28) srv_rerror_rate, 29) same_srv_rate, 30) diff_srv_rate,
31) srv_diff_host_rate, 32) dst_host_count, 33) dst_host_srv_count,
34) dst_host_same_srv_rate, 35) dst_host_diff_srv_rate,
36) dst_host_same_src_port_rate, 37) dst_host_srv_diff_host_rate,
38) dst_host_serror_rate, 39) dst_host_srv_serror_rate,
40) dst_host_rerror_rate, 41) dst_host_srv_rerror_rate, 42) attack_type.
```

For example, two records (# 1 and # 745) of File 1 are as follows:

```
0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,
0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,
normal.
184,tcp,telnet,SF,1511,2957,0,0,0,3,0,1,2,1,0,0,1,0,0,0,0,0,1,1,0.00,
0.00,0.00,0.00,1.00,0.00,0.00,1,3,1.00,0.00,1.00,0.67,0.00,0.00,0.00,
0.00, buffer_overflow.
```

File 1.1 has also been prepared with the same 51608 records of the same format just without the last parameter 42) attack_type.

File 2 contains 311079 records of the same format as in File 1.1.
File 1.1 and File 2 are the test data files.

Note that KDD archive does not indicate the correct types of attack for none of the records of File 2. The only available information on possible attacks is gathered in Tab. 1 (column 'Code' is the emulator's code of attack). Nevertheless, File 2 has been used to test whether the emulator is able to detect unknown intrusions, which had not been presented in the training data of File 1.

The results of training the emulator by File 1 are shown in Fig.1, where right-hand screen represents the initial population of cFIN after SVD (Start cells: $|W_0| = 51608$), while left-hand screen shows cFIN after apoptosis and immunization ($h_1 = 0.5$, $|W_1| = 783$). Total training time (for AMD 1.5 GHz) is 62 seconds including 8 s for the 1st stage (SVD) and 54 s for the 2nd stage (apoptosis and auto-immunization).

During the recognition of the records of File 1.1 and File 2, the emulator writes test results into the output file in the format: Record # - attack_type. For example, four records (## 744-747) with test results for File 1.1 are as follows (see also Tab. 2):

```
744 - normal.
745 - buffer_overflow. !!!
746 - buffer_overflow. !!!
747 - normal.
```

The emulator also shows on-line projection of any pattern to 3D cFIN (see bold skew cross in both screens) and write the recognition result on the bottom panel (see "Class: back !!!").

Test results in Tab. 2 correspond completely to the correct attack types (parameter 42) of File 1.

Table 1. Attack types

Code	Attack type	File 1	File 2	Code	Attack type	File 1	File 2
0	normal	+	+				
1	apache2		+	16	pod	+	+
2	back	+		17	portsweep	+	+
3	buffer_overflow	+	+	18	rootkit	+	
4	ftp_write			19	saint		+
5	guess_passwd		+	20	satan	+	
6	imap			21	sendmail		+
7	ipsweep	+	+	22	smurf	+	
8	land	+		23	snmpgetattack		+
9	loadmodule			24	spy		
10	multihop		+	25	teardrop	+	
11	named		+	26	udpstorm		+
12	neptune	+		27	warezclient		
13	nmap			28	warezmaster		
14	perl			29	xlock		+
15	phf	+	+	30	xsnoop		+

Table 2. Test results for File 1.1

Records ##	attack_type	Records ##	attack_type
745-746	Buffer_overflow	38036-38051	ipsweep
3095-7373	Smurf	38052-38151	back
9520-9523	Buffer_overflow	38302-38311	ipsweep
9590-9591	rootkit	42498-42519	ipsweep
9928-10007	neptune	42548-42567	ipsweep
10072	Satan	42593-42594	ipsweep
10320	phf	42706-42708	ipsweep
13340-13519	portsweep	42730-42761	ipsweep
13569	land	42762-42770	buffer_overflow
13845-13864	pod	42771-42772	land
16326-16327	pod	42773-43385	neptune
17446-37902	neptune	44451-44470	neptune
37929-37939	ipsweep	44800-48452	smurf
37959-37963	ipsweep	48453-48552	teadrop
38005-38012	ipsweep	All other	normal

Another test has been performed over File 2 to check whether the emulator is able to detect unknown intrusions, which had not been presented in the training data of File 1. The intrusion is treated as unknown if the projection of corresponding pattern to cFIN lies outside of the unit cube (according to Definition 1). The emulator has recognized 13 unknown intrusions as the following records ## of File 2:

```
417, 12674, 97891, 139795, 170498, 176201, 177958, 232570, 236975,
296561, 296657, 96796, 297658.
```

According to Tab. 1, any unknown intrusion can correspond to one of the following types of attack that had not been presented in the training data:

```
apache2, guess_passwd, multihop, named, saint, sendmail, snmpgetattack,
udpstorm, xlock, xsnoop.
```

The recognition time per record is 15.7 ms for both tests of File 1.1 and File 2. This time includes not only computations but mainly reading the record from test file, visualization of the recognition result (cFIN's projection of the pattern) in both screens of the emulator and writing the result into output file.

4 Conclusion

According to test results, cFIN reduces the storing patterns by 65.9 times using apoptosis and auto-immunization without any loss of accuracy of recognition. Although this increases the training time (from 8 seconds to 1 minute for AMD 1.5 GHz), nevertheless, more important is the decrease of the recognition time at least by 60 times per pattern by decreasing number of the stored cells of cFIN to be compared with recognizing pattern.

It is worth noting that so good performance of cFIN (error-free recognition with rather low training time) on the data of real-life dimension looks unobtainable for main competitors in the field of computational intelligence like artificial neural networks (ANN) and genetic algorithms (GA). According to the comparison in [25] and [26], cFIN trains by at least 40 times faster and recognizes by at least 2 times correctly than ANN and GA on the tasks of environmental monitoring and laser physics. These tasks have rather low dimension: 17×23×6 for ecological atlas and 19×5 for laser diode. Such drawbacks of ANN and GA become especially inadmissible for the task of intrusion detection with rather high dimension 51608×41 and more.

It is also worth noting that cFIN differs essentially from the negative selection algorithm (NSA). Actually, NSA aims to provide a set of detectors for self-nonself discrimination [7], [9], whereas cFIN guarantees a minimal set of "cells" for the correct recognition of any number of classes based on "cytokines". Apparently, this makes cFIN advantageous not only for the intrusion detection on-line [23] but also for medical oriented applications to simulate cancer specific apoptosis [10]. Moreover, cytokines modulate proliferation and apoptosis of thymic cells as well as intrathymic T cell differentiation that includes not only negative but also positive selection [21]. Therefore, cFIN also seems to be better suited for such kind of simulations.

Acknowledgement

This work is supported by EOARD under project # 017007 "Development of mathematical models of immune networks intended for information security assurance".

References

1. Ader, R., Felten, D.L., Cohen, N. (eds.): Psychoneuroimmunology. Academic Press, New York (2001)
2. Bay, S.D. The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Dept. of Information and Computer Science (1999)
3. Bunk, S. Signal blues: stress and cytokine levels underpin a provocative theory of depression. The Scientists, 25 August (2003) 24-28.
4. De Boer, R.J., Segel, L.A., Perelson, A.S. Pattern formation in one and two-dimensional shape space models of the immune system. J. Theoret. Biol. 155 (1992) 295-333
5. De Castro, L.N., Timmis, J. Artificial Immune Systems: A New Computational Intelligence Approach. Springer-Verlag, London (2002)
6. Dasgupta, D. (ed.): Artificial Immune Systems and Their Applications. Springer-Verlag, Berlin (1999)
7. Dasgupta, D., Krishna-Kumar, K., Wong, D., Berry, M. Negative selection algorithm for aircraft fault detection. Lecture Notes in Computer Science, Vol. 3239. Springer-Verlag, Berlin (2004) 1-13
8. Fisher, P.B., et al. mda-7/IL-24, a novel cancer selective apoptosis inducing cytokine gene: from the laboratory into the clinic. Cancer Biology and Therapy 2 (2003) S023-S037

9. Forrest, S., Perelson, A., Aleen, L., Cherukuri, R. Self-nonself discrimination in a computer. IEEE Symposium on Research in Security and Privacy. Oakland, USA (1994) 202-212
10. Goncharova L.B., Jacques Y., Martin-Vide C., Tarakanov A.O., Timmis J.I. Biomolecular immune-computer: theoretical basis and experimental simulator. The 4th Int. Conf. Artificial Immune Systems (ICARIS'05). Banff, Canada (2005) (accepted for publication)
11. Goncharova, L.B., Melnikov, Y.B., Tarakanov, A.O. Biomolecular immunocomputing. Lecture Notes in Computer Science, Vol. 2787. Springer-Verlag, Berlin (2003) 102-110
12. Hisada, M., et al. Potent antitumor activity of interleukin-27. Cancer Research 64 (2004) 1152-1156
13. Horn, R., Johnson, Ch. Matrix Analysis. Cambridge University Press (1986)
14. Human immune system inspires NASA machine-software fault detector. NASA Bulletin October 26 (2004) Ames Research Center
15. Igney, F. H., Krammer, P. H. Immune escape of tumors: apoptosis resistance and tumor counterattack. J. Leukoc. Biol. 71/6 (2002) 907-920
16. Jerne, N.K. The immune system. Scientific American 229/1 (1973) 52-60
17. Jerne, N.K. Toward a network theory of the immune system. Annals of Immunology 125C (1974) 373-389
18. Kourilsky, P., Truffa-Bachi, P. Cytokine fields and the polarization of the immune response. Trends in Immunology 22 (2001) 502-509
19. Kurzrock, R. Cytokine deregulation in cancer. Biomedecine & Pharmacotherapy 55/9/10 (2001) 543-547
20. O'Garra, A. Cytokines induce the development of functionally heterogeneous T helper cell subsets. Immunity 8 (1998) 275-283
21. Savino, W., Dardenne, M. Neuroendocrine control of thymus physiology. Endocrine Reviews 21/4 (2000) 412-443
22. Tarakanov, A., Dasgupta, D. An immunochip architecture and its emulation. NASA/DoD Conf. on Evolvable Hardware (EH'02). Alexandria, USA (2002) 261-265
23. Tarakanov A.O., Kvachev S.V., Sukhorukov A.V. A formal immune network and its implementation for on-line intrusion detection. The 3rd Int. Workshop Mathematical Methods, Models and Architectures for Computer Networks Security (MMM-ACNS'05). St. Petersburg, Russia (2005) (accepted for publication)
24. Tarakanov, A.O., Skormin, V.A., Sokolova, S.P. Immunocomputing: Principles and Applications. Springer-Verlag, New York (2003)
25. Tarakanov A.O., Tarakanov Y.A. A comparison of immune and neural computing for two real-life tasks of pattern recognition. Lecture Notes in Computer Science, Vol. 3239. Springer-Verlag, Berlin (2004) 236-249
26. Tarakanov A.O., Tarakanov Y.A. A comparison of immune and genetic algorithms for two real-life tasks of pattern recognition. Int. J. Unconventional Computing 1/3 (2005) (in press)
27. Tecchio, V., et al. IFN{alpha}-stimulated neutrophils and monocytes release a soluble form of TNF-related apoptosis-inducing ligand (TRAIL/Apo-2 ligand) displaying apoptotic activity on leukemic cells. Blood 103/10 (2004) 3837-3844
28. Wall, L., Burke, F., Caroline, B., Smyth, J., Balkwill, F. IFN-gamma induces apoptosis in ovarian cancer cells in vivo and in vitro. Clinical Cancer Research 9 (2003) 2487-2496

Examining Refuge Location Mechanisms in Intertidal Snails Using Artificial Life Simulation Techniques

Richard Stafford[1] and Mark S. Davies[2]

[1] School of Biology, University of Newcastle-upon-Tyne, NE1 7RU, UK
richard.stafford@ncl.ac.uk
[2] Ecology Centre, University of Sunderland, SR1 3SD, UK
mark.davies@sunderland.ac.uk

Abstract. High intertidal rocky shores are extremely stressful habitats. Marine snails in these habitats experience highly desiccating conditions, and they locate refuges such as crevices and form dense aggregations of individuals to reduce the effects of desiccation. This study investigates the mechanisms of refuge location in *Melarhaphe neritoides* using a simple set of rules to mimic the behaviour of each individual snail as a computer simulation. Chance interactions with other individuals, other individuals' trails and the crevices which form part of the virtual environment result in a mainly self-organised pattern of aggregations and crevice occupation which match real patterns obtained in laboratory experiments. Simulations where the following of trails is removed result in a poorer match to the experimental data, indicating the importance of trail-following in establishing these distribution patterns. The study shows that artificial life based models are a potentially useful tool in the investigation of rocky shore systems.

1 Introduction

Self-organisation of aggregations of individual animals is a common phenomenon in many biological systems from bacteria though to insects and vertebrates such as flocks of birds and shoals of fish (reviewed by [1]). Self-organisation of aggregation may even explain some patterns of aggregation in human society, such as traffic congestion [2], [3]. Aggregations of individuals may have important biological functions including reproduction or reducing risks of predation [4], [5]. Aggregation may also benefit individuals in a more subtle manner, enhancing communications or social interactions between individuals [6].

Aggregations arise in several ways. Firstly they can arise from individuals moving towards, or remaining in, an area where environmental benefits occur [7]. Secondly, they can arise through self-organisation, in areas with no environmental benefit [7], although the presence of the aggregation may modify the environment and form a positive feedback loop. In many cases the cause of aggregation can be attributed to a combination of both environmental benefit and self-organisation. For example, moist, shaded conditions house greater numbers of woodlice than dry sunny areas [8], but this aggregation of individuals is at least in part caused by differences in behaviour of individual woodlice between the two conditions.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 520 – 529, 2005.

On the high intertidal region of rocky shores, environmental conditions are extremely harsh [9]. Immersion may only occur on the highest spring tides and wave splash can be highly unpredictable resulting in highly desiccating conditions [10]. This region of the shore is inhabited mainly by gastropod snails of the family Littorinidae, and the most commonly occurring species on UK shores is *Melarhaphe neritoides* (L.). *M. neritoides* feeds on an epilithic biofilm of bacteria, algae and lichens [11], [12] and is normally found at low water inhabiting refuges, either inside crevices or pits in the rock surface or in dense aggregations consisting of up to 100 individuals [10]. Often aggregations occur inside crevices, or if the crevice is small, the aggregation may be centred on the crevice, with individuals spilling outside of the crevice [10].

Littorinid snails found in crevices and aggregations have body tissues with higher water content than those found individually on flat rock surfaces [13], [14], [15]. The temperatures of crevices and the evaporation rates from the crevices have also been shown to be lower than on flat rock [15]. Despite the clear advantages of occupying these areas, the mechanisms of locating crevices or forming aggregations are currently only speculative [16], [17], [18]. Experimental approaches to investigate the role of trail-following in producing aggregations have proved inconclusive [18] and invasive techniques such as the removal of mucus producing glands would greatly affect the behaviour and locomotive ability of snails and are therefore unsuitable for these behavioural investigations.

In this study we propose a technique to investigate how snails can locate refuges, such as crevices and aggregations, based on computer simulations of their individual behaviour and chance interactions with other individuals, other individuals' trails or with areas of environmental benefit (i.e. crevices). We show that an artificial life computer simulation technique can be used to examine refuge location mechanisms in intertidal snails. We investigate the importance of trail-following in the formation of aggregations, and we evaluate the role of self-organisation in the formation of aggregations relative to the role of a certain area's environmental benefit.

2 Materials and Methods

In this study the results of computer simulations are compared to results of laboratory experiments. In the experiment, a number of snails of the species *Melarhaphe neritoides* were placed on marble plates and immersed in seawater for 30 minutes. They were then removed and the plates allowed to dry and the snails to stop moving. The number of snails in aggregations of three or more individuals (three individuals have been used in previous studies as a measure of aggregation e.g. [10], [18]), where each individual was in direct physical contact with another, was determined, as were the number in crevices, which were formed by partially drilling holes of diameter 4 mm into the marble plate. Full details of the experimental methods can be found in [10], but in the current study a constant temperature of 22 °C was used and the number of snails placed on each marble plate was altered between replicate trials. The plates were cleaned between each replicate to remove previously laid mucus trails.

2.1 Simulating Individual Snail Behaviour

An individual snail of the species *Melarhaphe neritoides* was placed on a marble plate in a tank of freshly collected, aerated, seawater for 30 minutes and then removed from the water and allowed to continue moving whist the plate dried (all movement stopped in < 45 minutes). The movement of the individual was recorded using time-lapse photography and the data from 10 replicate snails statistically analysed to create a computer simulation of snail movement [15]. The snails showed three distinct movement phases, beginning and ending with tortuous movement, with a less tortuous pattern in the middle of the movement phase (Fig. 1a).

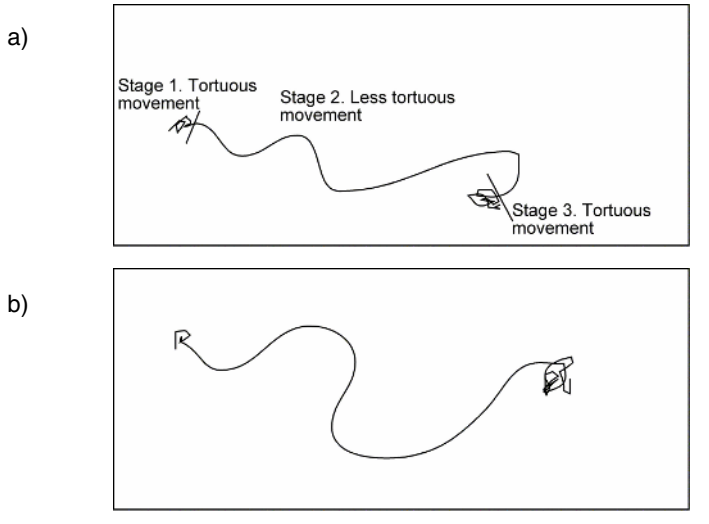


Fig. 1. Trails of the snail *Melarhaphe neritoides* moving on marble plates during 30 minutes of submersion and 45 minutes exposed to air. (*a*) an example of an observed trail. (*b*) an example of a simulated trail. Both trails show three stages of movement with an initial and final tortuous phase and a longer, less tortuous, middle phase.

The movement pattern was simulated on a grid of squares of side length 1.7 mm. This value was used because the mean distance between the centre of one square and a neighbouring square (including diagonal neighbours) was 2.0 mm which was the approximate length of each snail used to obtain the results. This allowed a movement of one body length in a real snail to equate to moving from one square to a neighbouring square in the simulation. Since little difference in the speed of movement of snails was observed (unless trail-following, see below) this distance can be defined as one timestep (t) in the simulation. The initial direction of movement was determined randomly by generating a number from a uniform distribution between 0 and 360 degrees. The first, tortuous movement phase occurred for a number of timesteps determined by a random number generated from a normal distribution of mean 15.1 and S.D. 2.4. For each of the timesteps in this movement phase an angle was generated from a normal distribution of mean 0 and S.D. 46.1. This angle was added to the angle at timestep t-1 to form a cumulative angle between 0 and 360. The

second, less tortuous movement phase was simulated with mean duration 114 (± 18.1 S.D.) timesteps, where the angle turned at each timestep was obtained from a normal distribution with mean value 0 and S.D. 12.3. The third, tortuous movement phase was simulated with mean duration 20.2 (± 2.4 S.D.) timesteps, where the angle turned at each timestep was obtained from a normal distribution with mean value 0 and S.D. 39.4. The cumulative angle determined to which neighbouring square the snail moved, with > -27.5 ° to 27.5 ° being vertically upwards, > 27.5 ° to 72.5 ° being diagonally up and right etc. A typical simulated movement pattern is shown in Fig. 1b to contrast with the real pattern from a snail in Fig. 1a. Statistical comparisons of 10 further real snail trails, compared with 10 simulated trails using fractal analysis of the trails (see [19] for details) showed no significant differences between simulated and real movement patterns.

2.2 Simulating Interactions and Decisions with Other Individuals, Trails and the Environment

Interactions occurred only through chance encounters. For example, if a snail encountered another snail by being in direct physical contact, either occupying the same square or a neighbouring (including diagonal) square, then an interaction occurred. If the snails were not in direct physical contact, for example in a nearest neighbour but one square, then, with the exception of trail-following (see below), there would be no interaction, unless the independent movement of the two snails resulted in them being in the neighbouring squares during a future timestep. Interactions with crevices occurred in the same manner, if a snail was in the same or nearest neighbour square as a crevice then an interaction would occur. For trails, the snail had to be in the same square as a previously laid trail to interact with it.

Once an interaction occurred, a decision was made regarding the interaction. If the decision was *successful* then the two (or more) individual snails which interacted would stop moving and remain in contact with each other, the snail would stop in a crevice or the snail would follow a trail. Further interactions between two or more snails that had stopped moving and an additional snail still moving could occur to form an aggregation (of three or more individuals) or to enlarge the size of an existing aggregation.

Successful decisions regarding trail-following resulted in the trail-following snail ignoring its normal movement patterns and following the trail of the trail laying snail in the direction it was laid (i.e. towards the trail laying snail). The movement speed of trail-following snails is faster than normal movement [15] and is simulated by the trail-following snail moving the distance moved by the trail laying snail in two timesteps in a single timestep. Trail following snails were still able to make decisions regarding crevice occupation or aggregation formation if interactions occurred. If the trail-following snail caught up with the trail-laying snail then the trail-following movement pattern stopped and an interaction between the two snails occurred.

Unsuccessful decisions of any kind resulted in snails continuing their individual movement pattern as if no interaction had occurred; however, the snails were prevented from making decisions about the same type of interaction for 10 timesteps. This prevention of the decision making process was necessary to prevent successive decisions occurring between the same individuals or individuals and environment in close succession. For example, each crevice could be located in 9 different squares, ten timesteps allowed the snail sufficient time to leave the vicinity of the crevice and

stopped decisions occurring at each timestep as to whether to enter the crevice or not, as a decision as to whether to enter the crevice had already been made once it was first encountered.

Observations of snails made during laboratory experiments into their refuge seeking behaviour (e.g. [10], [15]) showed that they could form aggregations and stop moving, enter crevices, or follow pre-laid trails (from observations made by time-lapse photography) at any time during the movement period of the snails. The probability, however, of a *successful* decision as defined above increased with the time spent moving, and in many cases was correlated with the dryness of the plates once removed from the water.

The decision processes were modelled by generating a uniform random number between 0 and 1 and comparing it to a probability function statistically generated from time-lapse photography and observational data on the number of *successful* and *unsuccessful* decisions made in discrete 5 minute periods [15]. The probability function produced an independent variable (probability) which varied between 0 and 1 with increases in the dependent variable (timesteps) of the model (see Table 1 for details of the probability functions for each parameter). If the random number was lower than the probability value then the decision was *successful*.

Table 1. The probability of successful decisions occurring varied with changes in timestep (t) of the model. The cubic equations below gave the best statistical match to the observed data collected where if a random number between 0 and 1 was lower than the value given by the equation a successful decision occurred. Where probability values fall below zero the probably is determined as zero and successful decisions were not possible at that timestep.

Behaviour	Probability of successful decision (p =)
Crevice	$(5.21 \times 10^{-2}) - (1.26 \times 10^{-2})t + (2.44 \times 10^{-4})t^2 - (8.04 \times 10^{-7})t^3$
Aggregation	$(4.91 \times 10^{-2}) - (9.99 \times 10^{-3})t + (1.81 \times 10^{-4})t^2 - (5.01 \times 10^{-7})t^3$
Trail following	$(9.79 \times 10^{-5})t^2 - (1.03 \times 10^{-2}) - (3.34 \times 10^{-3})t - (3.00 \times 10^{-7})t^3$

2.3 Overview of the Computer Simulation

All individual snails were simulated on a grid of 176 × 88 cells, mimicking the plate size used in the laboratory experiments. Crevices were generated in random positions on the plates, centred on a square in the grid. The simulated crevice did not extend beyond the boundaries of a single cell, but it could be detected by snails in neighbouring cells (see above). Movement of the simulated snail from the edge of the plate resulted in it continuing its movement on the corresponding opposite edge position (this was similar to the laboratory experiment where snails could crawl on both sides of a 150 × 150 mm plate suspended by two thin wires [10]).

All snails began moving at the same time ($t = 0$), and continued until their movement period was complete or until a *successful* decision involving an interaction with a crevice or other individual occurred. Details of the flow of the simulation are shown in Fig. 2.

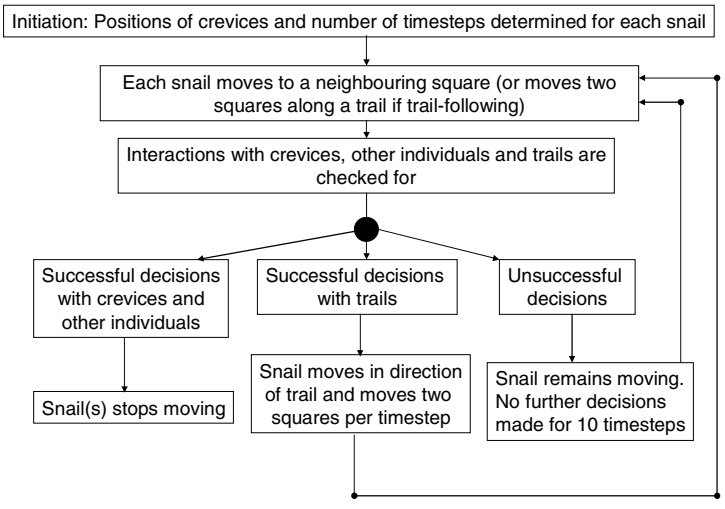


Fig. 2. Summary of the computer simulation. At each timestep snails move from one square on the grid to another. After movement of all snails, interactions are checked for and if any occur then decisions are made. The cycle continues until all snails stopped moving, either through making successful decisions about inhabiting crevices, through successful interactions with other individuals, or because their allocated number of timesteps in which to move has been reached.

3 Results

We systematically manipulated both snail and crevice density and measured either the percentage of snails in crevices or the percentage of snails in aggregations of three or more individuals. These measurements were taken for the laboratory experiments, a simulation where all trail-following was removed from the model and a simulation where trail-following was present. We performed separate trials to determine crevice occupation and aggregation formation to ensure the data was independent [20]. We performed 12 replicates for each combination of snail and crevice density for the laboratory experiments and both the computer simulations. In each case the mean value and 95 % confidence intervals were calculated (Fig. 3).

Crevice occupation was higher in computer simulations where trail-following was removed than in either the experimental data or the simulation including trail-following. With the exception of the 40 snails and 20 crevices treatment there was a high degree of conformity between the trail-following model and the experimental data (Fig. 3a). Increasing the crevice number from 20 to 30, with 20 snails, resulted in an increase in the percentage of snails occupying crevices (Fig 3a).

The percentage of aggregation formation did not increase significantly with snail density in either the experimental data or the trail-following simulation data, but differences occurred in the simulation with no trail-following (one way ANOVA

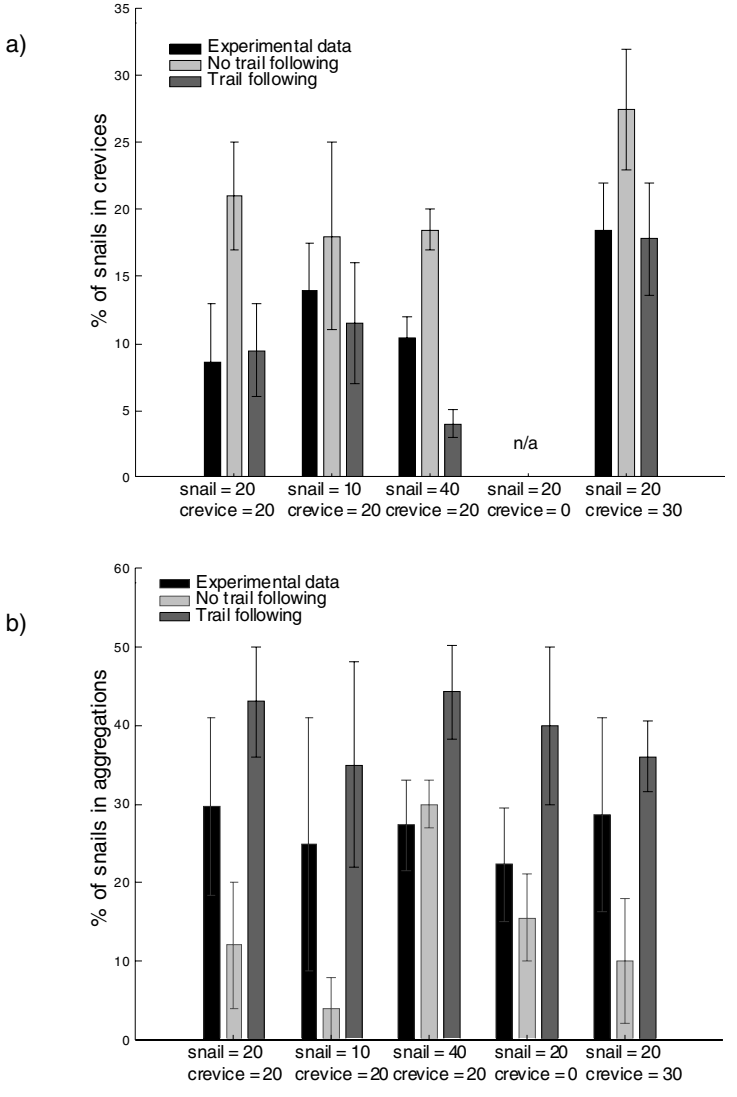


Fig. 3. The mean percentage (± 95 % C.I. n = 12) of snails in (*a*) crevices and (*b*) aggregations with differing numbers of crevices and snails on the plates. Data is given for the laboratory experiments, the non trial following model and the trail-following model.

investigating differences between 10, 20 and 40 snails with 20 crevices for experimental data $F_{2,33} = 0.05$ $p > 0.9$; for trail-following model $F_{2,33} = 0.42$ $p > 0.6$; for non trail-following model $F_{2,33} = 22.95p < 0.001$; see also Fig. 3b).

Importantly, the inclusion of trail-following in the simulation allowed aggregation to occur at or above experimentally observed values, even at low snail densities. The mean percentage of snails forming aggregations, however, was generally higher in the

trail-following simulation than in the experimental results. The manipulation of crevice density, with fixed snail density of 20, had no significant effect on the percentage of snails aggregating in any of the simulations or the experimental results (one way ANOVA investigating differences between 0, 10 and 30 crevices with 20 snails for experimental data $F_{2,33} = 0.64$ $p > 0.5$; for non trail-following simulation $F_{2,33} = 0.83$ $p > 0.4$; for trail-following model $F_{2,33} = 0.86$ $p > 0.4$; see also Fig. 3b).

4 Discussion

This study has shown that artificial life based simulations can be an important tool in the study of rocky shore ecology, particularly in the prediction of distribution patterns of intertidal snails from their individual behaviours. Individual-based computer simulations, where interaction occurs between the individuals, can closely mimic the collective behaviour of the assemblage of snails on marble plates. The use of these techniques is rare in rocky shore ecology; although important exceptions have occurred (e.g. [21] demonstrates clustering behaviour in molluscs, but using a more traditional mathematical technique; [22] demonstrates the use of cellular automata in simulating the patch dynamics of algae on the shore). Although traditional ecological modelling techniques using mathematical and statistical equations can prove a good predictive tool for large scale distribution patterns of organisms (e.g. [23] uses such a technique to examine the distribution patterns of red squirrels over a large geographic area), individual-based models can be used to investigate mechanisms used by individuals to form these patterns [24].

This study uses only three kinds of interactions between individuals and the environment and yet describes the distribution patterns of a snail assemblage well. We have not tested the response of the model to parameters other than those obtained by rigorous experimental and observational procedures. However, we show that only the three interactions simulated are required to predict the distribution patterns of snails. We also provide evidence that trail-following is an important mechanism in the formation of aggregations, particularly when the density of snails is low and the chance encounters of individuals are reduced. In fact the simulations with trail-following perform similarly to the experimental data in showing no affect of snail density on the percentage of littorinids in aggregations, yet in all cases the mean values of the simulations are slightly higher than the experimental results. This may mean the importance of trail-following may have been overestimated in the simulations. Creating the artificial life based simulations allows manipulative experiments to be performed on the simulated animals. For example, in this study we are able to prevent trail-following by the snails, which is impossible to effectively perform on real animals. We show that the results of the trail-following simulation are more closely related to experimental data on the percentage of snails in aggregations than are the non trail-following simulation results. Although it is possible that the mechanisms in the trail-following simulation are not identical to those used in real snails, this is perhaps the best evidence in establishing the importance of trail-following in aggregation formation in littorinid snails that has been documented, previous experimental and observational studies have proved inconclusive [18].

It is clearly demonstrated that there is greater conformity between the trail-following model and the experimental results than between the non trail-following model and the experimental results for both crevice occupation and aggregation

formation. Interestingly, the incorporation of trail-following appears to reduce the affect of locating a less environmentally stressful microhabitat such as a crevice. Although it is unclear if the artificial crevices did provide a reduction in stress to emersed individuals, they were selected for by the snails in the experimental results. On a plate with 20 crevices 14.1 mm^2 or 0.03 % of the plate area was occupied by crevices, so approximately the same percentage of snails should be found in crevices if no selection took place, where as in reality between 5 and 15 % of snails were found in crevices. The percentage of snails aggregating was unaffected by the crevice density. These data suggest that it is the self-organisation of aggregations that is important in establishing distribution patterns of snails on marble plates and not the presence of areas of lower environmental stress. This prediction of the simulations may not apply on real shores, where large aggregations of snails occur in cracks and crevices in the rock surface [10], [13], [15], [16], [17].

This study has several limitations in assessing the mechanisms used to locate refuges by *Melarhaphe neritoides* or other littorinid snails on real rocky shores. Of particular importance is the grazing patterns shown by *M. neritoides* on some shores in the UK, where it mainly limits itself to feeding inside a grazing halo only a few centimetres in diameter, which it normally shares with many other individuals [12]. The large variation in crevice density and crevice shape found between shores may also play an important role in shaping distribution patterns. Further work would be necessary to establish if the same behaviours simulated in this study can account for these distribution patterns found on real shores, or if the parameters of the simulation need to be adaptive depending, for example, on the crevice density of the shore. It is possible that the persistence of mucus trails on the shore would allow some sites, such as crevices, which are preferentially selected for compared to bare rock [15], [17], to become sites for aggregation, not only through the site offering a reduction in environmental stress to the animal, but also through self-organisation since a large number of persistent trails will lead to the crevice over a period of time [25]. The individual-based simulation technique proposed here could easily be modified, given suitable observational data, to test these theories and to provide insight into complex distribution patterns in a real environment.

Acknowledgements

We would like to thank the anonymous reviewers of the manuscript for their helpful comments. We would also like to thank Dr Mike Burrows for advice on an early draft.

References

1. Chowhury, D., Nishinari, K., Schadschneider, A.: Self-organized patterns and traffic flow in colonies of organisms: From bacteria and social insects to vertebrates. Phase Transit. **77** (2004) 601-624
2. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics, Phys. Rev. E **51** (1995) 4282-4286
3. Nagatani, T.: The physics of traffic jams. Rep. Prog. Phys. **65** (2001) 1331-1386
4. Krause, J., Ruxton, G.D.:Living in Groups. Oxford University Press, Oxford (2002)
5. Sword, G.A., Lorch, P.D., Gwynne, D.T.: Migratory bands give crickets protection. Nature **433** (2005) 703-703

6. Dunbar, R.I.M.: Determinants of group size in primates: A general model. Proc. Brit. Acad. **88** (1996) 33-57
7. Parrish, J.K., Edelstein-Keshet, L.: Complexity, pattern, and evolutionary trade-offs in animal aggregation. Science **284** (1999) 99-101
8. McFarland, D., Animal Behaviour, Third Edition. Longman, Harlow (1999)
9. McMahon, R.F.: Thermal tolerance, evaporative water loss, air water oxygen consumption and zonation of intertidal prosobranchs: a new synthesis. Hydrobiologia **193** (1990) 241-260
10. Stafford, R., Davies, M.S.: Temperature and desiccation do not affect aggregation behaviour in high shore littorinids in north-east England. Journal of Negative Results: Ecology and Evolutionary Biology. **1** (2004) 16-20
11. Fretter, B., Graham, A.: British prosobranch molluscs. Ray Society, London (1994)
12. Stafford, R., Davies, M.S.: Spatial patchiness of epilithic biofilm caused by refuge-inhabiting high shore gastropods. Hydrobiologia (in press)
13. Garrity, S.D.: Some adaptations of gastropods to physical stress on a tropical rocky shore. Ecology **65** (1984) 559 – 574
14. Chapman, M.G., Underwood, A.J.: Influences of tidal conditions, temperature and desiccation on patterns of aggregation of the high-shore periwinkle *Littorina unifasciata* in New South Wales, Australia. J. Exp. Mar. Biol. Ecol. **196** (1996) 213-237
15. Stafford, R.: The role of environmental stress and physical and biological interactions on the ecology of high shore littorinids in a temperate and a tropical region. Ph.D. thesis, University of Sunderland, Sunderland (2002)
16. Raffaelli, D.G., Hughes, R.N.: The effects of crevice size and availability on populations of *Littorina rudis* and *Littorina neritoides*. J. Anim. Ecol. **47** (1978) 71 – 83
17. Britton, J.C., McMahon, R.F., Hart, J.W.: Relationships between topography, substratum composition and surface temperature, and the spatial distribution of intertidal fauna on rocky shores of south-western Australia. In: Wells, F.E., Walker, D.I., Kirkman, H., Letherbridge, R. (eds.) The Marine Flora and Fauna of Albany, Western Australia, Western Australian Museum, Australia. 521-540
18. Chapman, M.G.: Variability in trail-following and aggregation in *Nodilittorina unifasciata* Gray. J. Exp. Mar. Biol. Ecol. **224** (1998) 48 – 71
19. Erlandsson, J., Kostylev, V.: Trail following, speed and fractal dimension of movement in a marine prosobranch, *Littorina littorea* during a mating and a non-mating season. Mar. Biol. **122** (1995) 87 – 94
20. Underwood, A.J.: Experiments in ecology. Cambridge University Press, Cambridge (1997)
21. Focardi, S., Deneubourg, J.L., Chelazzi, G.: How shore morthology and orientation mechanisms can affect the spatial-organisation of intertidal molluscs. J. Theor. Biol. **112** (1985) 771-782
22. Burrows, M.T., Hawkins, S.J.: Modelling patch dynamics on rocky shores using deterministic cellular automata. Mar. Ecol. Prog. Ser. **167** (1998) 1 – 13
23. Rushton, S.P., Lurz, P.W., South, A.B., Mitchell-Jones, A.: Modelling the distribution of red squirrels (*Sciurus vulgaris*) on the Isle of Wight. Anim. Conserv. 2 (1999) 111 – 120
24. Taylor, C.E., Jefferson, D.: Artificial life as a tool for biological inquiry. Artificial Life. **1** (1994) 1-13
25. Davies, M.S., Hawkins, S.J.: Mucus from marine molluscs. Adv. Mar. Bio. **34** (1998) 1-71

A Method for Designing Ant Colony Models and Two Examples of Its Application

Mari Nakamura[1] and Koichi Kurumatani[2]

[1] Research Institute for Cell Engineering (RICE),
National Institute of Advanced Industrial Science and Technology (AIST),
3-11-46 Nakoji Amagasaki, Hyogo 661-0974 Japan
[2] Information Technology Research Institute, AIST
tagami-nakamura@aist.go.jp
http://staff.aist.go.jp/tagami-nakamura/

Abstract. An ant colony shows collective behavior through signal patterns formed by individual ants communicating among themselves. In this paper I devise a method for designing ant colony model and apply the method to design two types of ant colonies, focusing on ant sensitivity to signals. In the first type, I design three foraging models (trail, attraction and desensitization), by modifying a simple foraging model repeatedly, changing ant sensitivity to recruit pheromone to improve foraging by regulating allocation of ants. Out of them, the desensitization model shows the best foraging efficiency as a result of balanced allocation and stable behavior. In the second type, I design a task-allocation model between foraging and mound-piling tasks using independent signals for each task. It shows weak interaction between these tasks.

1 Introduction

In previous studies [1], I proposed ant colony models that show macro-scale collective behavior as a result of the micro-scale behavior of individual ants through the formation of meso-scale signal patterns. In these models, many homogeneous ants, who respond to local cues in their environment, indirectly communicate among themselves with pheromone signals. I studied the interaction between the formation mechanism of signal patterns and the regulation mechanism of task allocation.

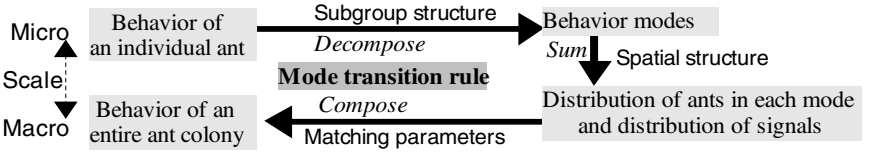


Fig. 1. Method for designing ant colony models

As illustrated in Fig. 1, I have devised a method comprised of following three steps to simplify the design of the collective behavior of an ant colony:

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 530 – 539, 2005.

1. Introduction of subgroup structure: Micro-scale behavior of an individual ant is decomposed into several behavior modes by using a mode-transition rule common to all ants in the colony. Ants in the same mode behave in the same manner.
2. Introduction of spatial structure: The distribution of ants in each mode is estimated from their behavior, integrating ants in the mode. The distribution of pheromone signals is calculated by the distribution of ants laying pheromone.
3. Matching parameters: Macro-scale behavior of an entire ant colony is constructed of the spatial distributions of both ants in each mode and the signal regions, using the same mode-transition rule.

Here, I apply the method to design the following two types of colony models, focusing on ant sensitivity to signals and introducing structures in mode transition rule:

1. In section 2, I design three foraging models (*trail, attraction and desensitization*), repeatedly modifying the simplest one by changing ant sensitivity to signals (by adding a new mode to the rule), to improve foraging efficiency.
2. In section 3, I design a task-allocation model between foraging and mound-piling using independent signals for each task (by combining two modules for each task in the rule), and investigate interaction between the two tasks.

2 Three Foraging Models: Trail, Attraction and Desensitization

In section 2, I apply the above method to foraging model of ant colony. When foraging, ants spread widely to search for food sources (*food-search subtask*), bring food to the nest while leaving *recruit pheromone* signals (*food-carry subtask*), or concentrate on the signals that lead the ants to the food source (*recruitment subtask*). In section 2.1, the recruit pheromone signals are explained. In section 2.2, I design the trail, attraction and desensitization models, repeatedly modifying the simplest model (trail) by changing ant sensitivity to the recruit pheromone, to improve foraging efficiency by regulating allocation to the above subtasks. In section 2.3, allocation of ants among these subtasks, stability of foraging behavior and foraging efficiency are compared among the simulation results of these foraging models.

2.1 Behavior of Signal Regions of Recruit Pheromone

An ant carrying a piece of food from the source to the nest leaves a trail of recruit pheromone on the ground. In these foraging models, the pheromone gradually evaporates and dissipates widely, as formulated in Eqs. 1 and 2 respectively, in which $T(x, y)$ denotes the density of the pheromone trail on the ground (x-y *plane:* $z=0$) and $P(x, y, z)$ denotes the density of the evaporated pheromone in the air.

$$(d/dt + \gamma_{vap})T(x, y) = 0 \tag{1}$$

$$\begin{aligned} \{d/dt - \gamma_{dif}(d^2/dx^2 + d^2/dy^2 + d^2/dz^2)\}P(x, y, z) &= 0 \quad (z > 0) \\ &= \gamma_{vap}T(x, y) \quad (z=0) \end{aligned} \tag{2}$$

As illustrated in Fig. 2, ants use two kinds of recruit pheromone signals: *trail* and *attracting area* defined as regions where $T(x, y)$ is stronger than T_{thr} and where $P(x, y, 0)$ is stronger than P_{thr}, respectively. Here, T_{thr} and P_{thr} denote thresholds.

Parameters used in simulations are listed as follows.

- Time constant $\gamma_{vap} = 0.21$, $T_{thr} = 0.01$ and amount of pheromone left by an ant in a step = *10.0* are determined to maintain a trail left by an ant for a while.
- The extent of attracting areas depends on diffusion factor $\gamma_{dif} = 0.42$ and $P_{thr} = 0.01$.

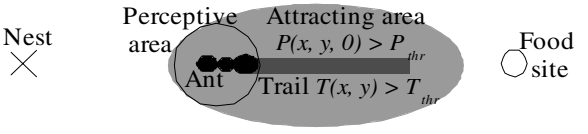


Fig. 2. Signal regions of recruit pheromone drawn by an ant carrying food

2.2 Designs of Three Foraging Models

Trail, attraction, and desensitization models are described in the following sections. They are simulated under the food-supply condition that "*after ants have consumed all the food units of one food site, a new food cluster of a predetermined size appears randomly within a 45 grid distance from the nest to maintain a constant number of existing food sites*". Parameters used in the simulations are listed as follows.

- Macro-scale parameters: Simulated space extends to *100 x 100 x 3* grids. In pheromone diffusion, the ground is a reflecting boundary, and the others are absorbing ones. The nest is located at the center of the ground. Colony is comprised of *600 ants* moving on the ground.
- Micro-scale parameters: One step is a period that an ant changes its action in response to a stimulus. Radius of the perceptive field = traveling distance of an ant in a step = *1.5 grid*. An ant "*walking randomly*" walks in a straight path and changes its direction randomly at *rate of 0.1 per step*.

Figs. 3, 4 and 5 indicate the mode transition rules of an ant, the dynamics among groups of each mode ants, and snapshots of simulation of the three models.

2.2.1 Trail Model: The Simplest Model

In this model, ants are only sensitive to trails. The colony is comprised of ants in *search*, *carry*, and *trace* modes (Fig. 3.1), which are allocated to food-search, food-carry, and recruitment subtasks, respectively. Their actions are defined as follows:

- A search-mode ant walks randomly. After finding a trail or food, it changes to trace or carry mode.
- A carry-mode ant leaves a pheromone trail while conveying one food unit from its source to the nest. When arriving at the nest, it drops the food unit into the nest, changes to trace mode and then randomly selects one of the trails surrounding the nest.
- A trace-mode ant follows the trail in the opposite direction from the nest. If it reaches or loses the food source, it changes to carry or search mode.

As Fig. 3.2 outlines, ants in the food-search subtask disperse, while the other ants concentrate on the trails. In this model, weak feedback loop among subtasks emerges, because the trails cannot recruit a sufficient number of ants. Ants devote most of their time to food-search subtask. As Fig. 3.3 displays, this model shows stable recruitment to all existing food sites with small short-term fluctuations and dominance of the food-search subtask in allocation.

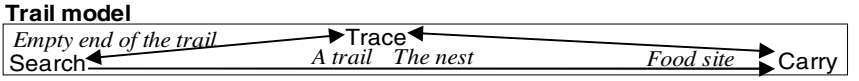


Fig. 3.1. Mode transition rule of an ant

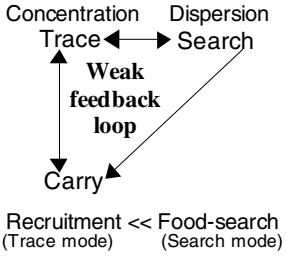


Fig. 3.2. Dynamics of model

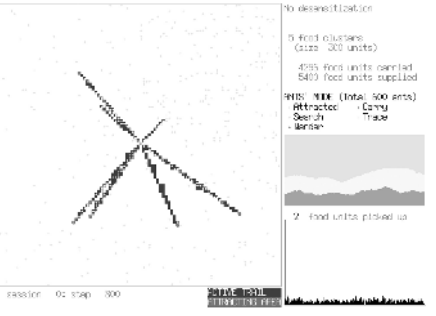


Fig. 3.3. Snapshot at step 800

Fig. 3. Explanation of trail model: Snapshots in Figs. 3, 4, and 5 are simulated with *300 units of food clusters* constantly supplied at *five food sites*. Left side of the frame in snapshot displays distributions of each mode ant (gray dots), pheromone signals (dark gray regions), the nest (black cross at the center), and food sites (large gray dots). Logs for ant rate in each mode (from top to bottom: wander, search, attracted, trace, and carry) and for the number of food pickers for the last *200 steps* are displayed on the right side of the frame.

2.2.2 Attraction Model: Intensifying Recruitment

In this model, ants are sensitive to both the trails and attracting areas. To recruit more ants from a wider area, an attracting area is introduced, and *attracted* mode is appended to the mode transition rule (Fig. 4.1). Its action is defined as follows:

- If a search-mode ant finds an attracting area, it changes to attracted mode.
- An attracted-mode ant moves to a point where evaporated pheromone is strongest within its perceptive area. If it reaches a trail, it changes to trace mode. If it loses the attracting area, it changes to search mode.

This model shows irregular and unstable long-term fluctuation caused by repeating deadlock (that is the state in which excessive concentration of recruited ants enclosed within signal regions suppresses search for new food, though the recruited ants wait for recruitment to new food until the signals disappear), as follows:

1. Search-decreasing stage (Fig. 4.3a): Recruitment to all food sites is observed. The rate of search-mode ants rapidly decreases because ants are caught within signal regions. The number of ants picking up a food unit (that is, *food pickers*) shows modest fluctuations.
2. Enclosure stage (Fig. 4.3b): Almost all ants are enclosed within a few signal regions, and this impedes the search for new food sources. Recruited ants move in a group following signals, quickly consuming marked food sites one after another. Then, the numbers of food pickers shows irregular, violent fluctuations.
3. Signal-vanishing stage (Fig. 4.3c): After deadlock occurs, signals soon vanish, and enclosed ants return to search mode and spread over the ground. The number of

food pickers is almost *0*. Search-mode ants soon find new food sites and switch to carry mode, repeating the above cycle.

As Fig. 4.2 outlines, mode transition of ants to search from trace or attracted mode is inactivated when enclosed within attracting areas, and feedback loop containing search mode is blocked. The mode transition from attracted to search mode is only active when the signal regions disappear after deadlock has broken.

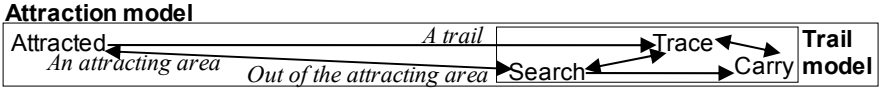


Fig. 4.1. Mode transition rule of an ant

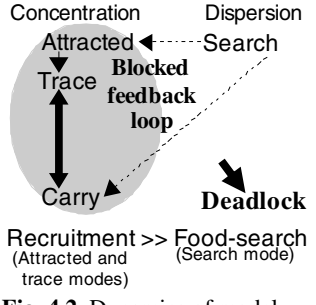


Fig. 4.2. Dynamics of model

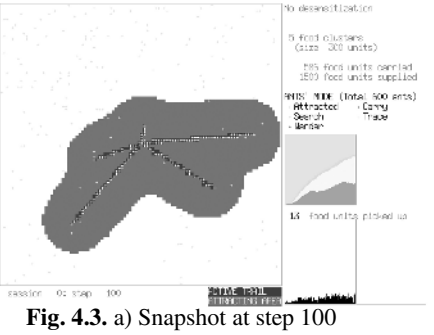


Fig. 4.3. a) Snapshot at step 100

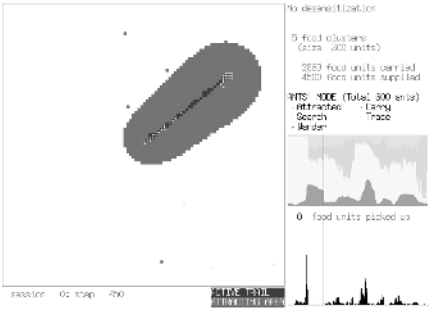


Fig. 4.3. b) Snapshot at step 450

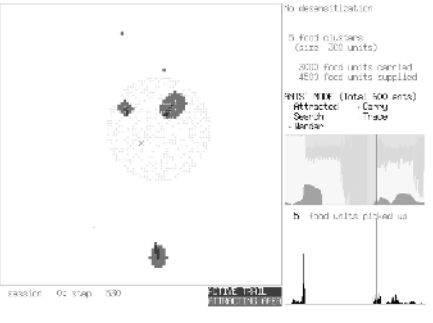


Fig. 4.3. c) Snapshot at step 530

Fig. 4. Explanation of attraction model: Details are explained in the legend of Fig. 3

2.2.3 Desensitization Model: Avoiding Deadlock

In this model, ants ignore pheromone signals for a certain period. To avoid deadlock, desensitization is introduced, and *wander* mode is appended to the mode-transition rule (Fig. 5.1). The action of a wander-mode ant is defined as follows:

- If a trace-mode ant reaches a trail end without a food source, it changes to wander mode.
- During a desensitization period, a wander-mode ant walks randomly, ignoring signals. After the period wears off, it returns to search mode.

The desensitization period is determined as *20 steps* to disperse wander-mode ants outside signal regions.

As Fig. 5.2 outlines, another feedback loop containing wander mode is created to circulate the ants among subtasks. The interaction between the localized concentrations of recruited ants on pheromone signals and long-range lateral inhibition emulated by widespread desensitized ants functions according to the similar mechanism to the reaction-diffusion system. As Fig. 5.3 displays, this model shows stable recruitment to all food sites with moderate fluctuations, and proper allocation between food-search and recruitment subtasks.

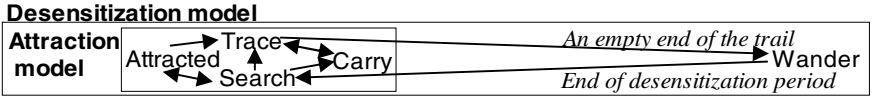


Fig. 5.1. Mode transition rule of an ant

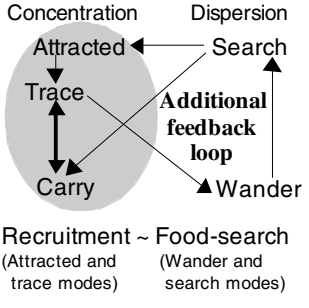


Fig. 5.2. Dynamics of model

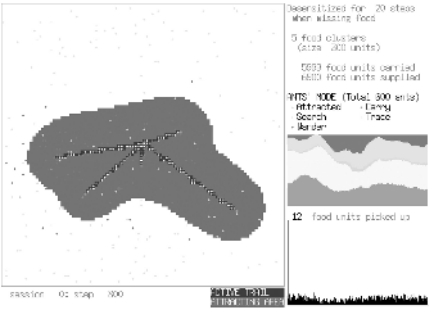


Fig. 5.3. Snapshot at step 800

Fig. 5 Explanation of desensitization model: Details are explained in the legend of Fig. 3

2.3 Comparison of the Three Foraging Models' Behavior

2.3.1 Allocation among Food-Search, Recruitment, and Food-Carry Subtasks

In Fig. 6, the trail model concentrates ants in the food-search subtask, and the attraction model concentrates ants in the recruitment subtask on time average. In contrast, the desensitization model shows proper allocation between food-search and recruitment subtasks and increase in allocation to food-carry subtask. The allocation to food-carry subtask which reflects foraging efficiency increases with proper allocation between food-search and recruitment subtasks, as argued in foraging theory [2].

2.3.2 Stability of Foraging Behavior

Fig. 7 shows time distributions from the appearance of a food site to its removal in each model. Compared with the others, distribution in the attraction model is promi nently scattered, showing strong correlation to distance from the nest because the enclosure of ants impedes the search for food sources far from the nest.

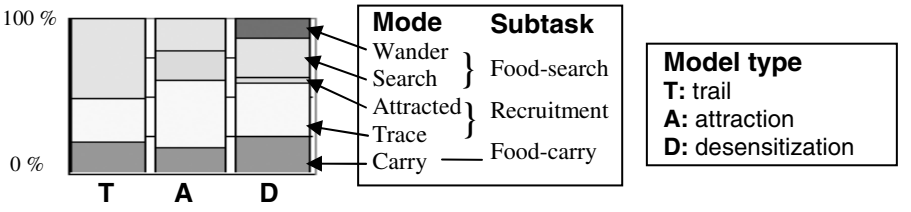


Fig. 6. Rate of ants in each mode in three foraging models, averaged for *5,000 steps*, simulated with *300 units* of food clusters constantly supplied at *five food sites*

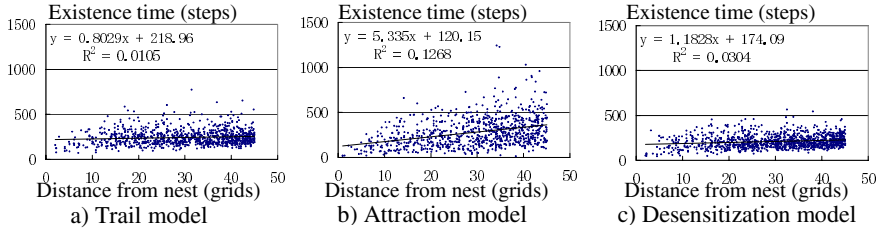


Fig. 7. Distributions of existence time of food sites in three foraging models, simulated under the same food supply condition as Fig. 6: Regression lines of the distributions and their residual sum of squares are indicated.

2.3.3 Foraging Efficiency with Varying Food Supplies

Fig. 8 shows the foraging efficiency of each model, with varying sizes of supplied food clusters and the number of food sites. The following features are indicated:

- As the size of food clusters and the number of food sites increase, foraging efficiency increases in all three models.

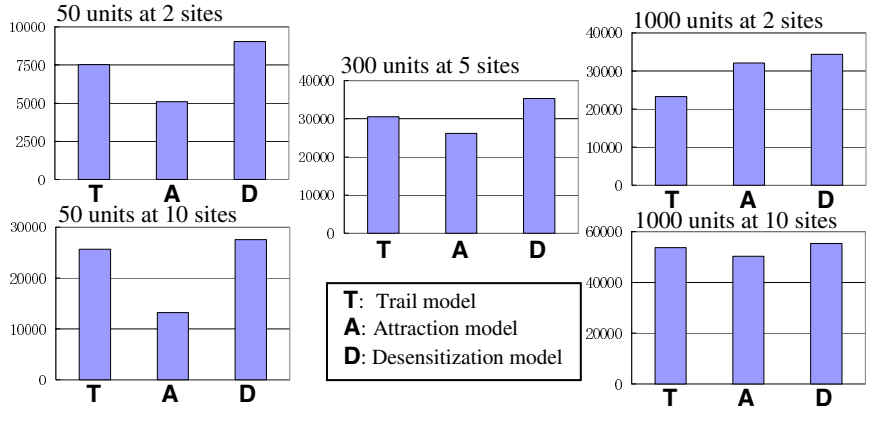


Fig. 8. Number of food units carried in each model at *5,000 steps* averaged across *10 simulations* with varying number of food sites and sizes of supplied food clusters

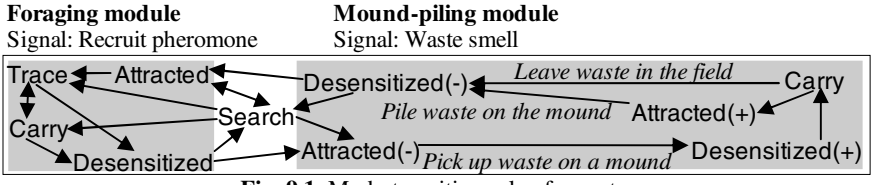


Fig. 9.1. Mode-transition rule of an ant

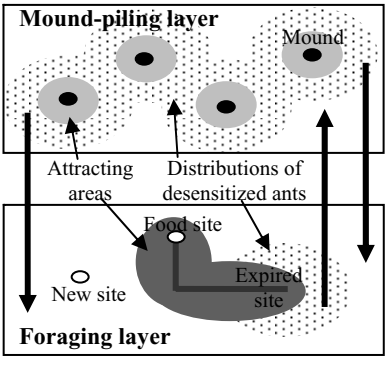


Fig. 9.2. Dynamics of model

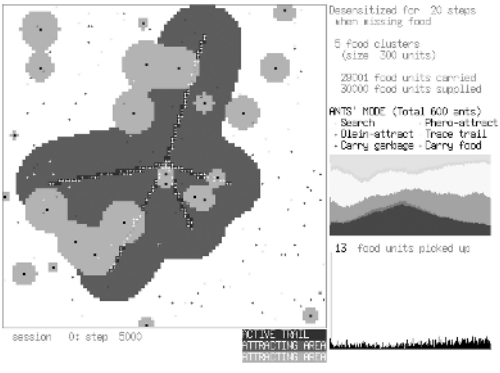


Fig. 9.3. Snapshot at step 5000

Fig. 9. Explanation of task-allocation model: Mode transition rule, dynamics, and snapshot of simulation of this model are shown as same as in Figs. 3, 4 and 5. Snapshot is simulated both with *300 units* of food clusters constantly supplied at *five sites* and with *1,000 waste units* piled at the nest at *step 0*. The left side of the frame in the snapshot displays mound distributions (large black dots), attracting areas of the mounds (light-gray regions, displayed over pheromone signals), and the others (the same tokens as that in Figs. 3). In the middle of the right side, the logs for the rate of ants in each mode for the last *200 steps* (from top to bottom; search- and desensitized-mode ants carrying nothing, attracted to recruit pheromone, trace-mode, carrying food, attracted to waste smell, and ants carrying waste) are displayed.

- The desensitization model demonstrates the best foraging efficiency among them, despite changes in the size of supplied food clusters[1] and the number of food sites.
- When the number of food sites is relatively small, the trail model's foraging efficiency is relatively low because trails cannot attract enough ants from wide areas.
- When relatively small size food clusters are supplied, the attraction model's foraging efficiency is relatively low because deadlock frequency increases.

3 Task Allocation Model Between Foraging and Mound-Piling

3.1 Design of Task Allocation Model

In section 3, I apply the designing method to a task-allocation model that carries out foraging and mound-piling tasks using independent signals for each task. Ants in this

[1] When simulated with far smaller food clusters, the existence time of food site is far shorter than that of pheromone signal left by an ant, and this causes unnecessary recruitment. When simulated with food clusters *less than 20 units* and *less than 4 units*, the best foraging efficiency is observed in the trail model and in *a model without pheromone signals*.

model exclusively perceive either of the signals. As illustrated in Fig. 9-1, mode transition rule of this model is comprised of two modules for each task connected mutually through the desensitized mode (which is similar to that in the other task-allocation models [3]), not to concentrate ants in either of the two tasks.

3.1.1 Appended Signals to Task Allocation Model: Waste Smell

When mound-piling, ants repeatedly pick up such waste as garbage or a corpse, and carry it to waste mounds. The waste smell evaporates from the mounds and dissipates into the air, as formulated in Eq. 3, in which $S(x, y, z)$ denotes the strength of the waste smell. The *attracting area of a waste mound* is defined as the region where $S(x, y, z)$ is stronger than threshold S_{thr}.

$$\begin{aligned} \{d/dt - \gamma'_{dif}(d^2/dx^2 + d^2/dy^2 + d^2/dz^2)\}S(x, y, z) &= 0 \quad (z > 0) \\ = \gamma'_{vap} \times &\textit{Number of waste units piled on } (x, y) \quad (z=0) \end{aligned} \tag{3}$$

Time constant $\gamma'_{vap} = 0.3$, diffusion factor $\gamma'_{dif} = 0.3$, and $S_{thr} = 0.1$ are determined to make the above signal both continue for several steps and expand for several grids.

3.1.2 Appended Mode Transition Rule: Modes in a Mound-Piling Module

The foraging module in Fig. 9-1 is the desensitization model in section 2. To introduce the mound-piling module, the corresponding actions are appended, as follows. Suffix $(_+/_-)$ indicates whether an ant of the mode has a waste unit or not.

- When a search-mode ant or "*a desensitized-mode ant in the foraging module*[2]" finds an attracting area of a waste mound, it changes to attracted($_-$) mode.
- An attracted-mode ant moves to a point where waste smell is strongest within its perceptive area. When reaching the mound, an attracted($_+$)-mode ant piles the unit on the mound [3] and switches to desensitized($_-$) mode, and an attracted($_-$)-mode ant picks up a unit from the mound and switches to desensitized($_+$) mode.
- A desensitized-mode ant walks randomly, ignoring waste smell. If a desensitized($_-$)-mode ant finds an attracting area of recruit pheromone, it switches to "*attracted mode in the foraging module*". After the desensitization period, a desensitized($_+$)-mode ant changes to carry mode, and a desensitized($_-$)-mode ant changes to search mode.
- A carry-mode ant walks randomly. If it finds attracting areas of waste mounds within a fixed waste carriage period, it switches to attracted($_+$) mode; unless it finds the attracting areas within the period, it leaves the unit there and changes to desensitized($_-$) mode.

The period of desensitization to the waste smell is also determined as *20 steps*. The waste-carriage period is determined as *70 steps*, on which the density of organized mounds depends. Parameters and conditions for foraging are the same as in section 2.

---

[2] This is a wander-mode ant in the desensitization model, as explained in Section 2.

[3] As an exception, ants are forbidden to drop wastes on the mound at the nest.

3.2 Simulation Results: Weak Interaction Between the Two Tasks

In simulations under the condition explained in legends of Fig. 9, the task allocation model indicates weak interaction between the two tasks through the distributions of desensitized-mode ants in the two modules (as outlined in Fig. 9-2), as follows:

- Pattern formation: Approximately *10~15* large mounds repeatedly appear and disappear slowly changing their distribution in response to removal of food sites. This is because ants desensitized to recruit pheromone are distributed around expired food sites, and remove wastes on the neighboring mounds away.
- Task allocation: The foraging efficiency of the task allocation model is fairly high despite a large allocation to the mound-piling task. Before and after removal of the waste mound at the nest[4], the foraging efficiency of the task allocation model is smaller than that of the desensitization model in section 2 by *20%* and by *16%* (averaged over *5 simulations*), although the proportion of ants allocated to the mound-piling task is *31%* and *28%*, respectively. Its mechanism is explained as follows. Widely dispersed waste-carrying ants become only sensitive to the recruit pheromone after carrying wastes. This compensates for food-search subtask in foraging.

4 Discussion

In section 2, desensitization is used to avoid deadlock, in the same way as decision error of ants in the colony model of Deneubourg [4]. In section 3, desensitization is used as a balancer among tasks, in the same way as task-allocation model of Gordon [3]. By using the designing method, more complex structures can be introduced in mode transition rule, changing ant sensitivity to signals.

5 Conclusion

I have devised a simple method for designing collective behavior, and have applied the method to the following two types of ant colony models.

1. The three foraging models comprised of ants with different sensitivities were designed. Among these models, the desensitization model shows the best foraging efficiency as a result of both proper allocation among subtasks and stable behavior.
2. The task allocation model between foraging and mound-piling, using independent signals for each task was designed. Weak interaction between tasks is observed.

References

[1] Nakamura M. and Kurumatani K. (1997) "Formation mechanism of pheromone pattern and control of foraging behavior of an ant colony", *In: Langton C. G. et al. (eds) Artificial Life V, The MIT Press*, pp 64-77
[2] Stephens D. W. and Krebs J. R. (1986) "Foraging theory", *Princeton University Press*
[3] Gordon D. M. (2000) "Ants at work", *W. W. Norton Company*, pp 143-147
[4] Deneubourg J.-L. et al. (1983) "Probabilistic Behavior in Ants; A Strategy of Errors?", *J. Theor. Biol.* 105:259-271

[4] In these simulations, it takes approximately *1,300 steps* to remove the nest mound away.

Simulating Artificial Organisms with Qualitative Physiology

Simon Hartley[1], Marc Cavazza[1], Louis Bec[2], Jean-Luc Lugrin[1], and Sean Crooks[1]

[1] School of Computing, University of Teesside, Middlesbrough, TS1 3BA United Kingdom
{S.Hartley, M.O.Cavazza} @tees.ac.uk
[2] CYPRES, Friche de la Belle de Mai, 41 rue Jobin F-13003, Marseille

Abstract. In this paper, we describe an approach to artificial life, which uses Qualitative Reasoning for the simulation of life within a 3D virtual environment. This system uses qualitative formalisms to describe both the physiology of a virtual creature and its environment. This approach has two main advantages: the possibility of representing integrated physiological functions at various levels of abstraction and the use of a common formalism for the simulation of internal (physiological) and external (environmental) processes. We illustrate this framework by revisiting early work in Artificial Life and providing these virtual life forms with a corresponding physiology, to obtain a complete living organism in virtual worlds.

1 Introduction

Previous work on Artificial Life has mostly considered molecular physiology, rather than higher-level physiological functions, with a few exceptions [9]. However, one of the challenges for the simulation of artificial life consists in being able to represent complex physiological functions to simulate organisms that are more complex. Our approach has been to use symbolic reasoning instead of differential equations and numerical methods to create a knowledge-based simulation of physiological functions. By this method, we are defining artificial physiology from first principles from a set of physiological processes, which opens new ways for the experimentation of artificial alternative life forms.

Using a symbolic description, a qualitative modeller could then devise a complex system that represents physiological phenomena from a library of common physical processes. The advantage of modelling such a system using Qualitative Reasoning is that using a suitably compositional approach for a model, allows the modeller to produce simple model fragments that combine to give the required complexity for the organism behaviour and produce results in real-time. The challenge in this method for model creation is choosing the best level of description and how to combine the model fragments that are produced from the analysis to create the world and phenomena for the artificial creature.

The system has been used to develop a virtual creature with internal processes and organs that react to changes in its environment. This work has evolved from research in A-Life that aimed at creating imaginary life forms [1, 2]. In addition to the creature, the environment has been simulated and integrated with the creature, which

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 540 – 550, 2005.

allows effects such as hot and cold currents, concentrations, and vortices to affect this creature through the qualitative simulations.

The creature we have created is an imaginary organism, which has a body comprised of organs that work together to carry on the various processes of life. Its main sustenance is extracted from the environment in which it exists and allows it to achieve homeostasis.

In the remainder of this paper, we will present the results from our ongoing research into the use of qualitative processes to simulate both creatures and "ecosystems" for artificial life within a 3D environment. We start by describing the software architecture for the virtual environment in which the virtual creature lives.

Following this are case studies into the visualisation of the processes within the virtual creature and its environment. In particular, we show how, due to the basic physical or physiological processes, we are able to instantiate multiple model fragments in the creatures' environment and have them interact with it. For example, we present an implementation of the artificial life form within a virtual "Ecosystem" as a test environment. This environment has been implemented as a fully immersive virtual reality system that the user can explore. In this virtual world, we have implemented various behaviours: for physical environment behaviour and for complex organ behaviour which are simulated in real-time. We conclude by discussion of the work completed so far and present our plans for future research into expanding the environmental effects into a self-contained ecosystem.

2 System Architecture

Our system is composed of two modules: a visualisation engine, which animates the virtual creature in its environment in real-time, and a qualitative simulation engine controlling the simulation of both the internal physiological processes of the creature, and of physical processes in its liquid environment (such as currents, heat flows, diffusion of nutrients, etc.). The integration of qualitative simulation with a 3D graphics system relies on the native event system of the visualisation engine. This event system has been extended to define high-level events that activate the QP simulations from the interaction with virtual world objects. We will refer to these events as Qualitative Physics events (abbreviated as *QP Events*). For example, when the creature enters or exits a volume, the event `enter_QPVolume` and `exit_QPVolume` are sent to the simulation. For instance the `enter_QPVolume` event can trigger processes of heat exchange between the creature and the liquid flows occurring in the volume it has just entered.

Our system is designed to operate both on standard desktop and within immersive Virtual Reality systems. The immersive system we use is a CAVE™-like system called an SAS-Cube™ the configuration of which consists of a four-sided PC-based hardware architecture that is powered by an ORAD™ PC cluster. It supports stereoscopic visualisation at 60 fps and real-time interaction. The QR Engine utilises the technique for qualitative simulation of physiological systems [4], derived from Qualitative Process Theory (QPT) [7] which we refer to as Qualitative Physiology. Qualitative Physiology represents the physiological processes governed by

physiological laws using the process-based formalism of QPT, and supports the real-time simulation of physiological sub-systems.

As the *QP Events* involve objects which are part of the simulation, they trigger the updating of relevant qualitative variables in the QP engine, hence prompting a new cycle of simulation.

In a similar fashion, to present the effects of the simulation to the visualisation engine we have devised *QP Effects*. These effects utilise the discretisation of the qualitative variables namely the "landmarks" and "limit points". When a qualitative variable passes either a landmark or limit point, a *QP Effect* is generated and sent to the graphical environment, to trigger changes in visual appearance corresponding to the landmarks reached. In addition, when the processes become active or inactive or when an object changes its *QPStates, QP Effects* are also generated. This can be used to produce a variety of visualisations, such as particle systems for fluid motion or colour changes for concentration. The software architecture for the communication of these *QP Events* and *QP Effects* utilises the UDP protocol for transmission between the qualitative simulation engine and the visualisation engine, [4,5,6]. Figure 1 shows an overview of the system architecture.

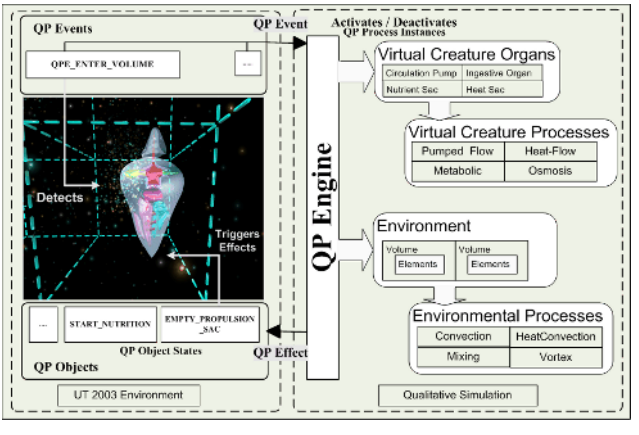


Fig. 1. System Architecture

The creation of the artificial creature includes the description of its anatomy and its physiology. The anatomical structure of the artificial creature is briefly outlined in Figure 2: Creature Physiology Overview. The visual contents have been produced using 3D modelling and animation packages such as 3D Studio Max™ and XSI™. These models as well as animations have been imported into the UT 2003 engine. These graphical representations can describe, using key-framed animation, the behaviours, and actions for the virtual creature. In our scenario, these animations represent movements of internal organs, changes in shape of the creature, as well as locomotion and since the simulation is controlled using *QP Events* we retain the interactive nature of the simulation.

We have described several physiological processes for the artificial virtual creature, dealing with elementary physiological functions such as nutrition,

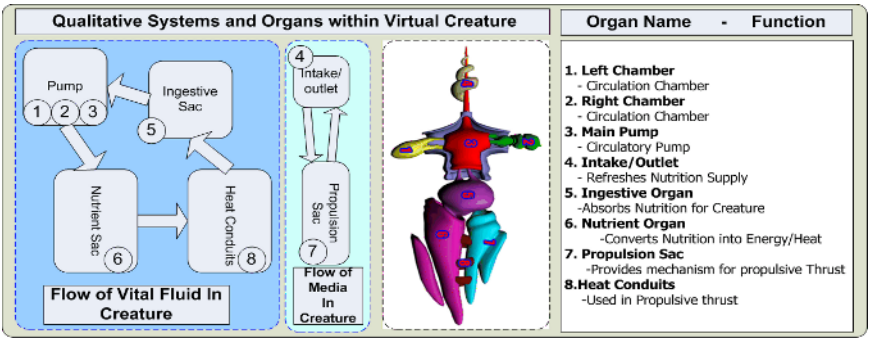


Fig. 2. Creature Overview

locomotion, and certain homeostatic processes such as thermal regulation. An essential aspect is that these are defined altogether as an integrated system, which can further be refined through experimentation, as the process description is highly modular and processes only connect through characteristic physiological variables (e.g., concentration in nutrients, temperature in certain organs, etc.). Defining the creature using this level of description is largely a functionalist approach [3], although a top-down, non-emergent one.

3 Implementation

As we previously introduced the Qualitative Simulation Engine utilises an artificial intelligence technique called qualitative process theory (QPT). Originally, this theory was developed for modelling complex mechanical and physical systems by abstracting physical descriptions of the phenomena [8].

The complete qualitative description requires us to use a method called envisionment by which we encapsulate the properties of the system and the relations between them. The qualitative description of the dynamics of the variables is described by a qualitative equation called influence equations. An example of this would be the influence equation for an osmotic system.

I+ (Amount–of Solute (Destination), A(OsmoticRate))
I- (Amount–of Solute (Source), A(OsmoticRate))

Here I+ represents the positive influence, and I- the negative influence, of the first value upon the second. We would in this case say the amount of solute in the *destination* is directly affected by the amount OsmoticRate. These equations constitute a declarative formalisation of the causal relations between qualitative variables. During the activation of the QP they determine the evolution of qualitative variables. The propagation of the effects of these equations creates an overall pattern of parameter changes. For instance, homeostasis within the creature is maintained by parameters, which regulate the rate at which the influences (*Processes*) act. An example of this is the increase in the metabolic rate to combat heat loss into the environment when the temperature of the creature is too low.

3.1 Physiology Implementation

The definition of the creature's physiology is based on a mapping between its organs and their physiological functions. In other words, the first step consists in designing the creature's anatomy with high-level physiological functions for each organ (e.g. heat conduit, or propulsion sac). In a second step, the detailed mechanisms behind these functions are described through the physical processes that constitute them. These functions are described though the *processes* and through qualitative states that are contained within Individual Views. Individual Views are a series of qualitative equations that are mutually exclusive and are used to calculate the qualitative state, the indirect influences and generate the effects that control the organ representation.

Using this method, we have composed the creature from a number of qualitatively modelled organs each of which has a number of Processes and Individual Views. The organ processes operate primarily upon the vital fluid that they contain and the physiology operates to create homeostasis within the properties of this fluid. A brief overview of the organs functions is shown in Figure 2: Creature Overview.

Of particular impact upon the creature’s homeostasis are the Nutrient organ and the Ingestive organ. These organs have, respectively an osmosis and metabolic process, which replenish and deplete the vital fluid of nutrients. Therefore, the contention between the operations of these *processes* is regulated by the creature to try to meet homeostasis. This is achieved by affecting the rate of vital fluid transfer between the organs, which is influenced by the Pump-Rate of the creature. The processes for the chemical (nutrient) homeostasis are combined with a thermal regulation system which completes the physiological systems for the creature. This thermal regulation allows the creature to combat the effects of heat-loss to the environment. The main detriment to the creatures’ temperature is the aforementioned heat loss. This is combated in part by the Metabolic process. The Metabolic process occurs within the nutrient sac and its effects (directs influences) are to remove Nutrient from the vital fluid in the organ and convert it into stored energy and heat. To achieve thermal equilibrium the creature may consume its store of energy to produce heat should its average temperature fall.

The description of an organs 'individual views' is used to encompass all the behaviour states for the organ. Each organ contains a number of individual views representing states for behaviours such as saturated, compressed or failure states such as depleted energy.

Our approach supports the data-driven propagation of behaviours throughout the various compartments of the creature's organ system. In Figure 3: Example Qualitative Description of a Propulsion Sac *QPState* and associated process, we see the qualitative definition of an organ state and an example of a process, which occurs within it.

This description of the object itself allows it to use its form to define its function. The processes within the propulsion sac are used to control the Sac when the creature is in its locomotion *QPState*. For instance, the propulsion sac filling / contracting processes open and close the valves, which control the filling and ejection of the propulsion sac.

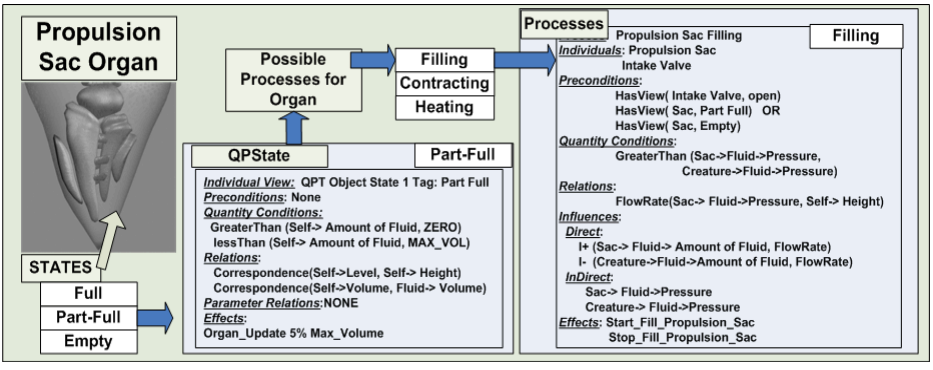


Fig. 3. Example Qualitative Description of a Propulsion Sac QPState and associated process

3.2 "Ecosystem" Implementation

In this Section, we explore the implementation of the ecosystem using qualitative formalisations for both the world physics and the creatures' physiology. Conceptually we have formed three levels of description for the ecosystem, which we label *environment*, *volume,* and *element*. These correspond to a hierarchical decomposition of the virtual space, each level being associated with different kinds of qualitative processes or variables. The highest level of the qualitative model for the environment is used to represent, in part, those behaviours that relate to the external world. For instance, one such interaction would be the heat dissipation from the environment volumes to an "external world" allowing a thermal equilibrium to be formed.

The environment is composed of "path objects" which form the interconnections for the volumes, defining a topology for the environment allowing exchanges between volumes, such as the flow of heat or particles. These "path objects" have as part of their composition, references to individual *volumes.* A "Path object" by its parameters control which of the processes act upon the referenced volume and the degree to which they act. For instance, the properties of the path objects are used by the convection process. In this case path-distance *(*a *spatial parameter* set by spatial separation for the environment*)* and conductivity are both used to determine the transfer rate parameter for the process.

The *volume* is composed of *elements*, which are particle-based representations of a fluid element within the special volume. Hence, volume parameters refer to its "microscopic" constituency in terms of elements: concentration, temperature, viscosity, etc. The *volume's* concentration is the key parameter used for the selection of its qualitative states. This parameter will be influenced, for instance, by convection currents within the environment. See Figure 4 for a description of the Convection process which occurs within a volume and the Description of the active state. The temperature of the *volume* and the viscosity of the *volume* increase as the concentration increases. The aggregation of volume properties (landmark values reached by the above *volume's* parameters) support the definition of global states for the volume. For instance, when the temperature is high and the concentration is high, the volume passes a limit point formed by the combination of these parameters and is said to enter a "perturbed" state, upon which the localised disturbance acts.

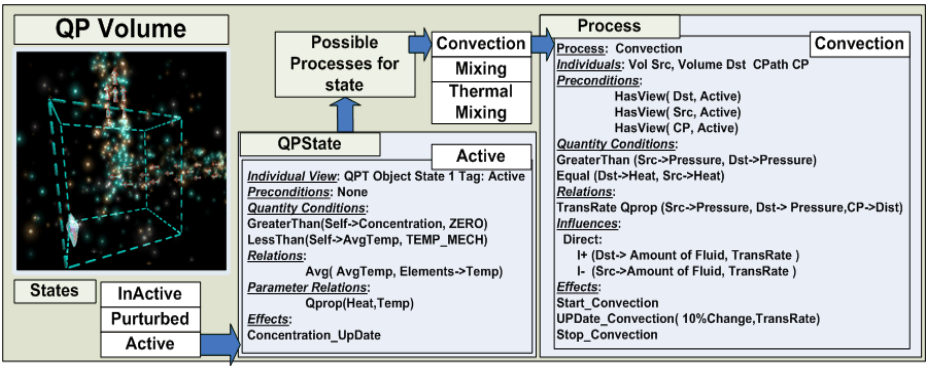


Fig. 4. Qualitative Volume State Description and Process

4 Results

The integration of qualitative physiology into the simulation for the environment has given us a unique avenue of investigation that allows us to develop experiments in a virtual ecosystem. This implementation allows the simulation processes to be altered to determine the survivability of the creature as a function of its physiology. A number of processes are active between the creature and its environment when the creature is in the normal steady state producing the required homeostatic responses. We have implemented a first prototype of the system in which we have a basic set of physiological and environmental processes. In this section, we illustrate the system behaviour by describing some specific results from the simulation.

4.1 The Locomotion Scenario

The integrated approach to the simulation has allowed us to develop simulation effects for the environment as well as for the creature, which has allowed the triggering of different individual view states for the creature dependant upon the conditions that are prevalent in the environment. The major individual view states for the creature are used to control a locomotion system. When the locomotion is active the creature utilizes its own system for propulsion, in which the organ states change, and the creature expends its internal stored energy from its metabolic processes to power thrusts from the propulsion sac, which give it a hoping motion through the environment.

The normal operation for the creatures' physiology involves the transfer of vital fluid around the creatures' organs whose transfer rate is dependant upon the parameter "pump-rate", which is a property of the circulatory pump organ. The nutrients are carried within the vital fluid and are replenished by the osmosis process, within the ingestive organ, and depleted by the metabolic process, within the nutrient sac. The rate of depletion in the metabolic process is given by the organs' parameter

"conversion rate". The metabolic process depletes the nutrients in the vital fluid and coverts them into heat and stored energy.

The locomotion *QPState* changes the relations between these parameters increasing the overall rate of the creatures' physiology. For instance, the "pump-rate" parameter within the circulatory pump organ becomes dependant upon the creatures' parameter "stored-energy" in away such that the parameter increases toward its maximum the further the amount of stored energy deviates from its maximum. The decrease in "stored energy" is due to consumption of the energy by the heating conduits process, which is activated by the new *QPState* and converts "stored energy" into heat.

The locomotion state changes the metabolic process, which is active in the nutrient sac, by increasing the conversion rate. Thus, the main effect of the new state is to activate the propulsion sac and the heat conduits processes and to alter the entities normal physiological system allowing the processes to occur faster. The heat conduits and the propulsion sac organs comprise the locomotive system for the creature and their processes heat fluid for expulsion/propulsion and manage the system of valves for the organ respectively.

During the Locomotion state, the flow of sea media due to the pumping operation of the intake/outlet organ (Figure 2: Creature Physiology Overview organ 4), that replenishes the nutrients within the shell of the creature from the environment, is used also as a channel for propulsive thrusts. The media within the creatures shell we label internal media. This internal media can be drawn into the Propulsion Sac organ (figure 2. organ 7) via a valve on the organ. This filling of the organ is controlled by the propulsion sac expansion process, which sends the effect `QP_Effect_Fill_Propulsion_Sac` to begin the manipulation of the propulsion organ. The Sac closes the valve when full and activates the Heat Conduits (Figure 2. organ 8) which in turn heat the sea media producing a pressure increase.

When the pressure inside the Propulsion Sac passes a value it triggers the pressure valve, an expansion valve whose state depends upon the pressure, on the sac. The change in valve state activates the propulsion sac contraction process and the sea media is expelled from the Sac into the shell. The organ enters a contracting state which stops the action of the Heat Conduits and as the valve has been released starts the empting of the organ via the Propulsion Sac Contraction Process. This process directly affects the size of the Sac which indirectly affects the volume of the sac. The decrease in the size also affects the amount of media in the organ. The effects of this stage are shown in Figure 5: Propulsion Sac Organ Pressure. The propulsion sac contraction process generates the *QP Effect* `QP_Effect_Empty_Propulsion_Sac`, to the graphical environment, which changes the representation for the propulsion organ to deflating. The shell responds to the sudden increase by expelling fluid into the environment via an expulsion using the intake outlet organ creating a propulsive thrust. This expulsion process generates the `start_thrust` which visualises the thrust within the environment by a particle effect whose rate depends upon the rate of fluid flow. Figure 5: Propulsion Organ Pressure depicts these changes in the creature with a plot of the pressure within the organ.

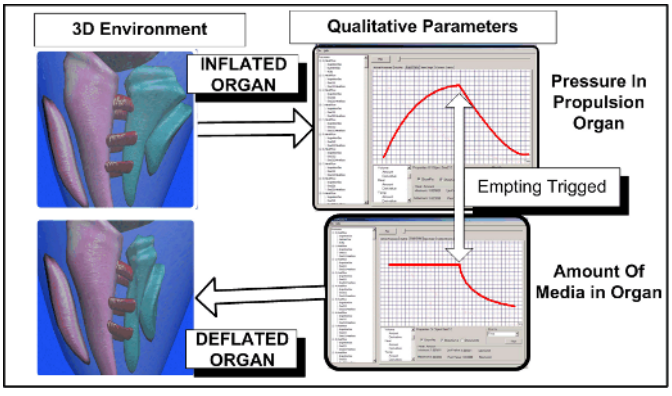


Fig. 5. Propulsion Organ Pressure

4.2 Environment Localised Perturbation Scenario

The environment is composed of an imaginary media whose properties are devised to react to give thermo-mechanical effects. The vortex is a product of this effect and is created by a combination of effects upon a volume.

A volume can either have nutrients, be depleted of nutrients or have an environmental effect active. These concepts correspond to the *QPStates inactive*, *active* and *perturbed* for a *volume*. A nutrient depleted volume (*inactive*) has no *QP Effects* as it has no graphical effects present and minimal process activity between its *elements*.

A nutrient rich volume (*active*) contains a high concentration of nutrients and thus we have chosen to represent an attractive volume within the environment with a distinctive effect. This is achieved by generating the *QP Effect* `Start_Nutrition` when the *volume* enters this *QPState*. This effect relates the parameter "concentration" within the *volume* to the lifetime of the particle within the 3D environment. This creates the effect of a denser cloud of particles for higher values of concentration. Also within this *QPState* as the value evolves through the effects of convection processes an Update_Nutrition *QP Effect* is generated for appreciable changes (~10% Saturation value). Figure 6: Environmental Processes (a). The *active* state allows the convection process to occur which is visualized by the sprite emitters that represent moving media Figure 6: Environmental Processes (b). Starting the process sends the *QP Effect* `Start_Convection_Process` to the 3D environment. This effect starts the convection sprite emitter at its maximum rate, as the process starts at its maximum and works to achieve equilibrium between the two volumes and the "path object" associated with the convection process provides the direction for the flow.

The media of the environment has been designed to respond to the presence of high concentrations of nutrient and heat; this response takes the form of a localized perturbation within the volume. When these conditions are met the properties at different points within the volume are evaluated and a pressure gradient between the bottom and top of the volume and the opposing sides of the volume is created. When

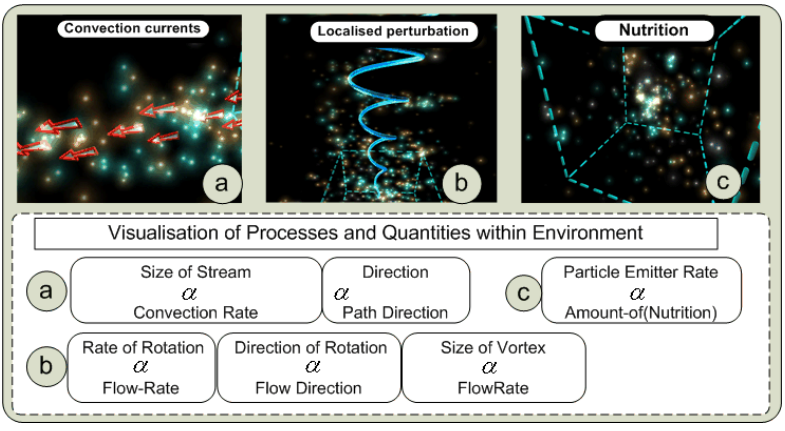


Fig. 6. Environmental Processes

a *volume* enters into a perturbed state it generates the `Start_Vortex` QP *Effect* that stops the current animation within the volume and allows the vortex process to begin which acts to restore the balance between the elements within the volume. This process calculates the rotation direction and rate using the properties of the volumes elements and the 3D environment uses this data to generate a vortex object Figure 6: Environmental Processes (c).

This leads to scenarios in which the creature can find patches of high nutrient concentration but by its presence cause instability in the volume, creating a vortex that drives the creature, either by its effect or by locomotion (if the creature has stored energy) from the volume.

5 Conclusion

This unique approach to integration of a creature's physiology and environment immerses the user within the 3D environment, modelled in real-time and provides a platform to approach new experimentations within artificial life. The system itself provides a successful attempt at real-time simulation of artificial life, which includes an imaginary physiology that interacts with the environment. The framework allows for the construction of homeostatic systems within the artificial creature that responds to influences from the environment. The environment itself provides the elements for basic creature behaviour such as a chemotactic response. Future work in this area will include the expansion of the creatures' perception of the environment, interaction between creatures and we are currently working upon linking the symbolic description of qualitative processes to evolutionary programming principles. For the environment future work will include expansion of the states of the volume to allow for different "cloud formations" for the nutrients dependent upon the parameters of the volumes elements instead of the single particle emitter which we are using.

Acknowledgements

This work has been funded in part by the European Commission through the ALTERNE project (IST-38575).

References

1. Bec, L., 1991. Elements d'Epistemologie Fabulatoire, in: C. Langton, C. Taylor, J.D. Farmer, & S. Rasmussen (Eds.), *Artificial Life II*, SFI Studies in the Sciences of Complexity, Proc. Vol. X. Redwood City, CA: Addison-Wesley (in French).
2. Bec, L., 1998. Artificial Life Under Tension: A Lesson in Epistemological Fabulation, In: C. Sommerer and L. Mignonneau (Eds.), Art @ Science, New York: Springer Verlag.
3. Bedau, M.A., Philosophical Aspects of Artificial Life, in: F.Varela and P. Bourgine (Eds.), Towards a Practice of Autonomous Systems, Cambridge: MIT Press, pp. 494-503. (1992)
4. Cavazza, M. and Simo, A., 2003. A Virtual Patient Based on Qualitative Simulation. In ACM Intelligent User Interfaces 2003, Miami, FL, USA, pp. 19-25
5. Cavazza, M., Hartley, S., Lugrin, J.-L. and Le Bras, M., 2002. Alternative Reality: A New Platform for Digital Arts, ACM Symposium on Virtual Reality Software and Technology (VRST2003), pp. 100-108, Osaka, Japan, October 2003.
6. Cavazza, M., Hartley, S., Lugrin, J.-L. and Le Bras, M., 2002. Alternative Reality: Qualitative Physics for Digital Arts, Proceedings of the 17th international workshop on Qualitative Reasoning 2003, Brasilia, Brasil.
7. Collins, J. and Forbus, K Building qualitative models of thermodynamic processes. *Proceedings of the Qualitative Reasoning Workshop*. (1989).
8. Forbus, K.D., Qualitative Process Theory, Artificial Intelligence, 24, 1-3, pp. 85-168.(1984).
9. Grand, S., Cliff, D. and Malhotra, A., Creatures: artificial life autonomous software agents for home entertainment, In: *Proceedings of the first international conference on Autonomous agents*, pp. 22-29. ACM Press. (1997)

Slime Mould and the Transition to Multicellularity: The Role of the Macrocyst Stage

John Bryden

School of Computing, University of Leeds, Leeds LS2 9JT
johnb@comp.leeds.ac.uk

Abstract. The transition from unicellular to multicellular organisms is one of the mysteries of evolutionary biology. Individual cells must give up their rights to reproduction and reproduce instead as part of a whole. I review and model the macrocyst stage in slime mould (*Dictyostelium*) evolution to investigate why an organism might have something to gain from joining a collective reproduction strategy. The macrocyst is a reproductive cartel where individual cells aggregate and form a large zygotic cell which then eats the other aggregating cells. The offspring all have the same genetic code. The model is a steady state genetic algorithm at an individual cellular level. An individual's genetic code determines a threshold above which it will reproduce and a threshold below which it will join a macrocyst. I find that cycles in food availability can play an important role in an organism's likelihood of joining the macrocyst. The results also demonstrate how the macrocyst may be an important precursor to other cooperative behaviours.

1 Introduction

The quest to synthesise hierarchical levels of organisation in artificial life is a significant open problem [3,23]. To provide a deeper understanding into how we may be able to use evolutionary algorithms to generate and optimise hierarchical behaviour, we can study the major transitions in evolution [16]. This work focuses on the transition to multicellularity which appears to be one of the most difficult 'bridges' evolution has had to cross. It is unclear whether the transition only occurred once, or several times [4]. Phylogenetic evidence [2] suggests that multicellular organisms, especially metazoa, share a common ancestor. Furthermore, fossil evidence [16] indicates that multicellular life did not exist for 2,500 million years until the Cambrian period (approximately 540 million years ago) where all the multicellular phyla are represented.

Multicellular organisms essentially consist of clusters of individual cells with all cells expressing the same genotype. They therefore require gene-regulatory mechanisms for differentiating cells (with differentiations being passed from parent cell to offspring), cell adhesion and spatial patterning of cells [16]. One particularly crucial cell differentiation stands out: The organism must separate its reproductive (*germ-line*) cells from its body (*soma*) cells [7].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 551–561, 2005.

The requirement for isolation of the germ line from the soma was first argued to be necessary by August Weismann [7]. To identify why, we can distinguish the two types of reproduction that are present in metazoan multicellular life and look at the conflicts that arise between them. Firstly, intra-organism reproduction happens when cells replicate within the super-organism, for the good of the super-organism. Conflicts can occur with cells reproducing on their own behalf [17]: mutant cells can disrupt and compete with the super-organism. By generating a whole organism from one initial germ-line cell, it is clear that the vast majority of selfish mutations that disrupt super-organism-level processes will only survive one generation [7]. Therefore, secondly, to solve this problem super-organism reproduction involves the replication of the complete organism through the selection of a germ line cell to reproduce on behalf of the super-organism. However, there is still a conflict over which cell is to be the germ line since selfish mutations that disrupt the super-organism reproductive process will be passed onto the next generation. A stable, policed, germ-line/soma differentiation mechanism must have evolved at some point.

It is unclear where in the evolution of a multicellular lineage, stable, well policed, germ-line/soma differentiation and germ line isolation should occur. However, given the above problems faced with intra-organism conflicts[17], it seems likely that the germ-line/soma differentiation evolved early [7]. Thus, we consider evolutionary mechanisms that will explain a transition between unicellular organisms, which compete within their populations and compete with predators and prey, and early multicellular organisms which are clustered together and exhibit germ-line/soma differentiation. In other words, there is a transition from unicellular organisms which are optimised to maximise their own *direct* fitness to cells that must, on the other hand, maximise their *inclusive* fitness at the expense of their direct fitness (i.e., their ability to contribute their fitness to other cells that are highly related must be more important than their own replication chances). (See [10] for precise definitions of *direct* and *inclusive* fitness.)

Whether the evolutionary transition described above, of organisms clustering and differentiating a germ line, happened in one stage is unclear. Wolpert has presented a model where individual cells may split to produce a somatic body cell that sticks to its parent and is unable to reproduce [27]. What the benefits, through inclusive fitness, are to individual cells and their lineages from doing this is unclear. There is a debate on this subject with some arguing that size is an important reason for multicellularity [4] with undifferentiated population clustering, as modelled in [19] without a germ-line/soma differentiation, being an important first step. Others point out that local competition over food will negate the value of cooperation through relatedness [21,26,15]. For this reason Di Paolo warns against relatedness being used as an explanation for cooperative behaviour [9]. There therefore appears to be something of a paradox if we attempt to try to understand the transition to multicellularity with such models of clustering cells. Individuals that cluster compete with each other and may negate the benefits of cooperation through relatedness, yet both clustering and cooperation are needed for the transition to early multicellularity.

A different perspective considers multicellularity through aggregation [16]. Here cells either vegetate and reproduce individually, or aggregate to reproduce collectively. This presents a sort of half way house between the individual and early multicellular behaviour identified above. *Dictyostelium* (more commonly known as slime mould) is a model organism for multicellularity through aggregation [16,20]. Individual cells can either vegetate and reproduce asexually on their own, or under different environmental conditions they also demonstrate collective reproduction behaviour, characterised by individual cells making sacrifices for the benefit of other cells' reproductive chances. This organism therefore demonstrates both the germ-line/soma differentiation [6] and clustering that is important for the transition. Biological evidence is now presented concerning *Dictyostelium discoideum*, one of the more studied species of the genus.

When there is a shortage of food and *D. discoideum* cells begin to starve, they aggregate and one of the two collective reproductive stages commences [22]. The more well known reproductive stage of *D. discoideum* sees the cells form a slug which collectively migrates. Once the cells find an advantageous location they form a *fruiting body*: cells at the front of the slug (20%) form a stalk and the rest form spore cells at the top of the stalk which are dispersed by the wind. Interestingly, the stalk cells die after the stalk is built. This differentiation between spore and stalk cells is arguably a germ-line/soma distinction [6]. Since cells that produce stalks do not pass on their genetic code, it is hard to see how this trait is selected for and maintained. Indeed there are examples of slime moulds strains that do not produce stalks [6]. Computer simulations addressing this question [1] have indicated that high dispersal of spores can lead to more stability in the stalk producing behaviour.

The second, less well known, collective reproduction stage in *D. discoideum* involves the formation of the *macrocyst* [22]. Again, when the cells are starving they aggregate. However instead of forming a slug, two cells merge to form a large *Zygote* cell which eats other aggregating cells. The resulting giant cell forms a hard cellulose outer wall and this macrocyst germinates after a few weeks. See Fig. 1 for a diagram.

The macrocyst stage is thought to be a precursor to the slug/stalk reproductive stage. Kessin [13] argues that evolution generally occurs in incremental stages. He notes that the previous stage to macrocyst development would be the microcyst stage (not observed in *D. discoideum*), where individuals form outer walls on their own. After the evolution of chemotaxis, aggregation could occur and the macrocyst evolved. With added cell adhesion and cell type differentiation into stalks and spores, fruiting body and slug behaviour would then become plausible.

The genetic makeup of the offspring of the macrocyst is an important question. The macrocyst is generally accepted to be the sexual phase of *D. discoideum's* development [22]. However experiments do demonstrate that Macrocysts can form from only one mating type [5]. The progeny of one macrocyst is observed to be of one genotype [25]. Only one nucleus remains in the zygote (or giant cell) after other ingested nuclei disappear [18].

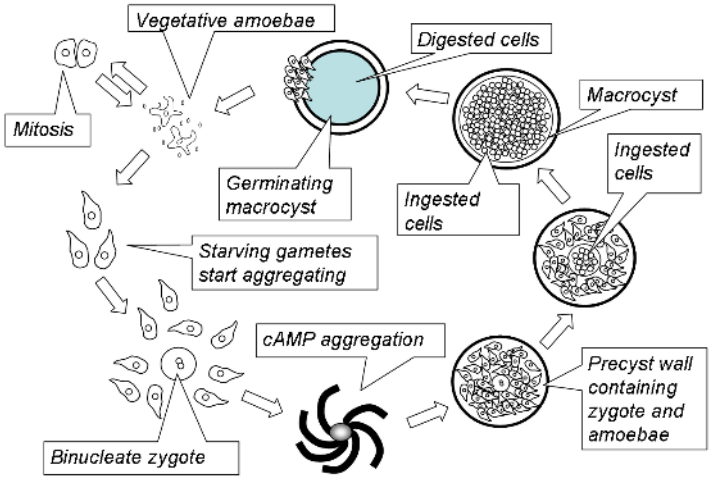


Fig. 1. The sexual and mitotic life cycles of *Dictyostelium* (based on [22])

From Fig. 1 it is clear that differentiation in *D. discoideum* cells occurs when it starts to aggregate. Recent evidence implies that the cell may have genetic control over this event. Research [8,11] suggests genes that can control or delay when or whether a cell will continue to grow or start aggregation. These findings indicate that the cell is capable of turning on or off aggregation to the macrocyst stage which can ultimately lead to cells being eaten by the zygote. This emphasises a need for an explanation as to why an individual might make the 'choice' to aggregate and almost certainly die.

I have produced a model of the *D. discoideum* macrocyst stage for several reasons: (i) to confirm that individuals that normally reproduce on their own are indeed prepared to gamble their own reproductive chances against the 'pot' of reproductive material contained in the macrocyst; (ii) to confirm my intuition that fluctuations in food availability are important to the viability of the macrocyst; (iii) to question the role individual mitotic split rates might play in the stability of the macrocyst; and (iv) to speculate on the role the macrocyst might play in the evolution of other altruistic behaviour (such as stalk/spore differentiation) and collective behaviour.

2 Methods

To investigate the questions in Section 1 I have built a computer simulation model of the macrocyst stage of *D. discoideum*. Assumptions in the model are based on the biological evidence presented. Notably I have assumed that all the offspring of a macrocyst are of the same genotype. Since sexual fusion does not seem to be necessary, I chose (on parsimonious as well as biological grounds) to model the macrocyst with no sexual recombination. Individual vegetative

behaviour was modelled with individuals having a genetically encoded energy threshold above which they mitotically reproduce.

D. discoideum cells are modelled as individuals in a non-spatial environment. At each time step, a number of individuals (N) are selected at random, each receives a 0.5 units of energy (representing food) with probability p. One cycle in the model contains two seasons. The amount and probability of food (N and p) changes value according to whether the season is 'high' ($N = 100, p = 0.6$) or 'low' ($N = 20, p = 0.3$). Each season lasts 200 turns. All individuals pay a daily energy cost ($E_c = 1.0$) irrespective of season. If an individual's energy falls below zero ($x < 0$), it will die.

Each individual cell is modelled with two genes[1] The genes model energy thresholds which determine the behaviour of the cell. Cells will join the macrocyst when their energy level is *below* the first gene, the *macrocyst join threshold* ($-2.0 < G_{\text{join}} < 2.0$). When a cell's energy level is *above* the second gene, the *split threshold* ($5.0 < G_{\text{split}} < 20.0$), the cell will pay an energy cost to split mitotically (see Fig. 1) and produce a new cell (sharing energy equally between itself and its offspring).

There is only one macrocyst in the model it is assumed to be immobile and therefore does not receive food from the environment. When cells join it, they contribute their own energy (x) plus a residual energy amount (equal to the cost of splitting) to the macrocyst's 'pot' (X). Before closing the macrocyst pays a cost E_m per individual joined every turn to reflect metabolisation and building of cellulose. If the macrocyst energy falls below zero ($X < 0$) then it (and all its joining cells) will die. When the macrocyst reaches a predetermined energy threshold (30.0), it closes and no other cells may join.

The macrocyst will germinate on the first turn of the high season. When it germinates, the energy is divided up into new cells with each cell receiving 2.5 energy units. All new cells will have the same genotype: a complete genotype (no recombination) is picked at random from all the cells that originally joined the macrocyst.

Simulations were run over 100,000 turns. Each simulation started with 100 individuals, each individual having a random genotype and a random energy between 0.0 and 5.0.

3 Results

To understand how the harshness of the low season can effect the viability of the macrocyst, simulations were run varying the probability of food in the low season. Interesting population dynamics, with macrocyst offspring out-competing the non-joining population, were observed and these are presented in this section.

The average percentage of individuals which germinated from the macrocyst is plotted against the probability of food in the low season in Fig. 2. When

[1] Genes are represented as floating-point numbers in the simulation, point mutations occur at each time step over a gaussian distribution with standard deviation of 1% of the gene space.

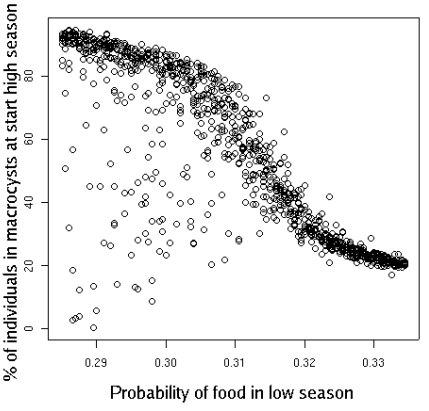


Fig. 2. Graph showing the percentage of individuals which germinated from a macrocyst at the start of the high season against the probability of food in the low season. Each data point (ten data points, each generated with different random seeds, per food-probability value) represents an average over a complete simulation run.

the probability of receiving energy is higher, few individuals ($\approx 20\%$) join the macrocyst. When there is a lower probability of food, more individuals join the macrocyst. However the rogue data points at the bottom left of the graph are of interest.

To investigate this disparity with some populations producing macrocysts and others not, the probability of food and seed value were selected from one of the rogue data points. The simulation was run over a longer (150,000) number of turns. A histogram was generated for the macrocyst join threshold at the start of each high season and the results are shown as a 3D mesh in Fig. 3A.

In the figure, the presence of macrocysts can be seen as spikes on the right hand side. An early tendency towards macrocyst joining is evident (far right of graph) but these genotypes die out after ≈ 25 cycles. A population which does not produce germinating macrocysts immediately flourishes. After ≈ 150 more cycles there are enough individuals to successfully produce a germinating macrocyst which survives to the end of the low season. Interestingly once this has happened the macrocyst very quickly wipes out the non joiners from the population. The offspring from the macrocyst must have some sort of competitive advantage over the non-joining population.

A closer look at Fig. 3A indicates that when there are not enough individuals joining the macrocysts to make them germinate, there is only a small tendency toward individuals that will not join the macrocyst when their energy is very low. Between cycle 25 and cycle 175, the histogram shows a larger proportion of individuals having a join threshold below zero, however some still remain with a threshold above zero. There is clearly little selection pressure against individuals sacrificing small amounts of energy when near to death.

A second 3D histogram was generated for the split thresholds of the population at the start of the high season and can be seen in Fig. 3B. There is a

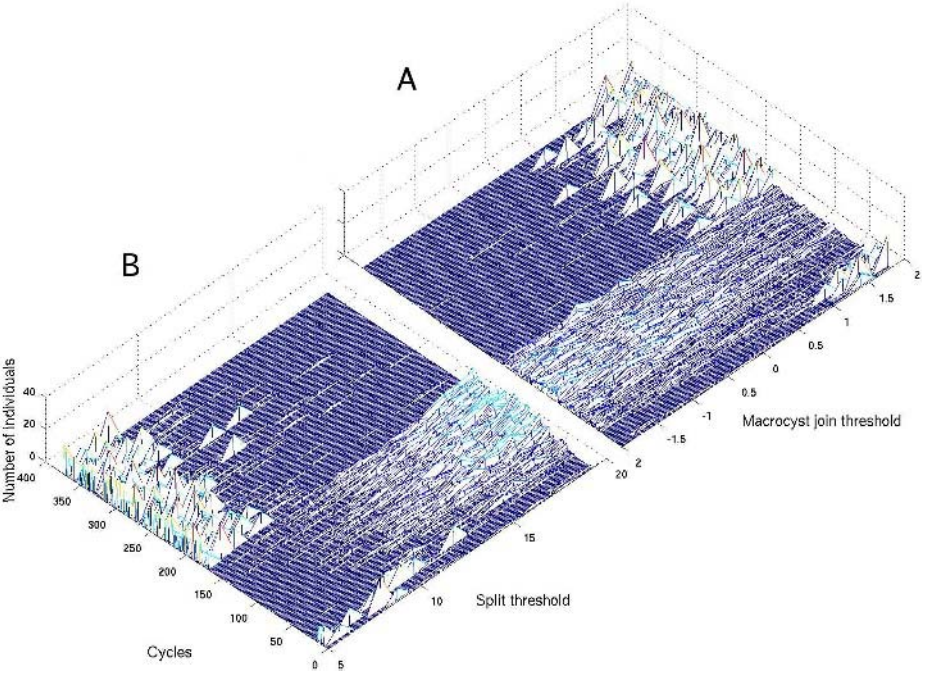


Fig. 3. 3D histograms of macrocyst join thresholds (A) and individual split thresholds (B) of the population at the start of each high season

clear disparity of the split thresholds between the macrocyst joining population and the non joiners. Again, in the first few cycles of the simulation (where the macrocyst joiners were predominant in Fig. 3A), the population has a low split threshold – individuals will split as quickly as possible. After ≈ 25 cycles the macrocysts die out. There is now a clear tendency for dominance in the population for individuals that split more slowly. Once the macrocysts return (after ≈ 160 cycles), the split thresholds of the population immediately return to lower values (< 7).

Simulations run with all individuals having the same, fixed, split threshold resulted in either the individuals all dieing, through starvation in the low season, or a small percentage joining the macrocyst when food is more plentiful (data not shown). The competitive advantage of the macrocyst joining population was no longer effective and macrocysts were only formed through enough individuals sacrificing their energy in a similar way to the non-joining population in Fig. 3A.

Other simulations have been run with variable split thresholds and the low season completely removed to see if parameters exist where a macrocyst can form and dominate the rest of the population. Simulations were run with varying parameters of N and p, both seasons having the same values. While some macrocyst production was observed it was only at the beginning of simulations

where the random starting population allowed for enough individuals that joined the macrocyst and made it viable for a few cycles (data not shown).

4 Discussion

In Section 1 I have argued of the need for a model that demonstrates the transition between individual cells that ordinarily reproduce on their own to cells that become part of a super-organism, with only one genotype of the participating cells being passed on to future generations. For the macrocyst model to successfully meet the requirements of this transition, it requires that all individual cells must be able to reproduce on their own. It also requires that individual cells must be clustered and that only one of the individual cells reproduces on behalf of the cluster. The model presented meets these requirements. Simulated cells that have the freedom to evolve a strategy in which they will not join macrocyst organisations (where their genes are highly likely to be destroyed) do not evolve this strategy under fluctuating environmental conditions.

The model does however stop short of demonstrating the type of germline/soma differentiation and clustering apparent in the metazoa where there is differentiation of the germ line early in development [17] and permanent clustering (as in other models, e.g., [27,19]). The macrocyst's germ-line cell is the zygote which is not differentiated from any other cells in the super-organism. Also, the macrocyst cells are only clustered at one point of the life cycle. However, the fact that the macrocyst's offspring are of only one genotype and that they out-compete individuals that do not join the macrocyst is of some significance.

The fact that the macrocyst produces offspring of a single genotype is important in three ways. Firstly it has the effect of producing several homogenous offspring which are all 'preprogrammed' to join the macrocyst at the start of the next low season. These offspring have a competitive advantage over individuals that do not join the macrocyst. The macrocyst therefore contributes to its future success. Since microbes can evolve many 'policing' mechanisms [24], it is not inconceivable that after several generations, the macrocyst way well have become established in the organism without the need for a harsh low season each cycle.

Secondly, the high relatedness of the offspring can be seen to promote other social behaviours. Relatedness is crucial for any traits that require many coordinated individuals or altruism to be successful. The aclonal nature of the macrocyst offspring means that it is highly likely that the next aggregation event will also be aclonal or at least highly related. If these individuals have the same mutation which means (perhaps under certain environmental conditions) they no longer fuse to form a zygote then other interesting collective behaviour may occur instead. These behaviours could include, but are not limited to, the slug behaviour of *D. discoideum* which requires many coordinated individuals [14], and the stalk behaviour of *D. discoideum* which requires altruism from many cells [1]. The macrocyst has been argued to be a precursor of these behaviours [13]. The combination of the macrocyst model with one of the

stalk/spore behaviour (based on [1]) will hopefully confirm how important the population homogenisation effects of the macrocyst were for the evolution and maintenance of stalk/spore behaviour in *D. discoideum*.

The homogeneous macrocyst offspring are important in a third way: By picking the genotype of its offpsring from one individual at random, the macrocyst stage eradicates the potential for cheating: If an individual were to evolve a 'cheating' trait so that its genes were most likely to be picked, the next population would all have that same trait - with no individual having any advantage.

To consider how split thresholds are important I analyse a complete cycle. In one cycle of the model presented here there are four phases for non macrocyst joining amoebae: (i) Early high season exponential growth; (ii) Population equilibrium at high season; (iii) Early low season exponential decimation of the population; (iv) Population equilibrium at low season. While it is easy to see that fast (low threshold) splitting amoebae would flourish during phase (i), these same amoebae will be closer to dying during phase (iii). The results suggest that a slow (high threshold) splitting strategy is more profitable, not only in phase (iii) but in phase (iv) as well. In phase (iv) individuals receive food with a low probability, those with a fast (low) split threshold are less adapted to survive fluctuations in food availability. The macrocyst allows individuals to avoid phases (iii) and (iv) and hence fast splitting individuals that germinate from it at the start of the high season are very well adapted to phase (i). This ability to perform well during circumstances of diminishing populations has already been observed as an important feature of early multicellular organisms [12].

While I have attempted to be faithful to biological evidence, the model presented here has made some assumptions and has some limitations. Further analysis and research is required into the biological plausibility of the split thresholds in the model. The question as to what might happen if individuals have a seasonally varying split threshold is also important. The model is undimensional and therefore lacks spatial effects (though the way the organisms are fed is set up to mimic a spatial environment): a spatial model would allow us to analyse what might happen if individuals could effect their chances of being the chosen genotype. The mutation rate in the model is unnaturally fast, however slower mutation rates provided similar results over longer periods. Finally there is only one macrocyst in the current model, future simulations will model more than one macrocyst.

The model and results presented in this paper demonstrate that, given the assumptions outlined, the *D. discoideum* macrocyst stage is plausible under the large fluctuations in food in the model. The results and analysis lead me to hypothesise that the model of the macrocyst presented in this paper, where individuals gamble their genes to become the germ line of a super-organism, may well have been a crucial stage in the transition to multicellularity. It must be noted that it is only a stage in the evolution of *D. discoideum* and may be relevant only to this organism. However, the facts that the slug behaviour of *D. discoideum* is reminiscent of other metazoa and that their phylogeny

implies a common ancestor imply that slime mould may give some important clues into the evolution of the metazoa and perhaps other multicellular organisms.

Acknowledgements. Thanks to Jason Noble and Richard Watson.

References

1. D. P. Armstrong. Why don't cellular slime molds cheat. *Journal of Theoretical Biology*, 109:271–283, 1984.
2. S.L. Baldauf, A.J. Roger, I. Wenk-Siefert, and W.F. Doolittle. A kingdom level phylogeny of eukaryotes based on combined protein data. *Science*, 290:972–977, 2000.
3. M. A. Bedau, J. S. McCaskill, N. H. Packard, S. Rasmussen, C. Adami, D.G. Green, T. Ikegami, K. Kaneko, and T. S. Ray. Open problems in artificial life. *Artificial Life*, 6:363–376, 2000.
4. J.T. Bonner. The origins of multicellularity. *Integrative Biology*, 1:27–36, 1999.
5. D. M. Bozzone and J. T. Bonner. Macrocyst formation in *Dictyostelium discoideum*: Mating or selfing? *The Journal of Experimental Zoology*, 220:391–394, 1982.
6. L. W. Buss. Somatic cell parasitism and the evolution of somatic tissue compatibility. *Evolution*, 79:5337–5341, 1982.
7. L. W. Buss. *The evolution of individuality*. Princeton University Press, 1987.
8. S.C. Chae, Y. Inazu, A. Amagai, and Y. Maeda. Underexpression of a novel gene, *dia2*, impairs the transition of *Dictyostelium* cells from growth to differentiation. *Biochemical and Biophysical Research Communications*, 252:278–283, 1998.
9. E. Di Paolo. A little more than kind and less than kin: the unwarranted use of kin selection in spatial models of communication. In D. Floreano, J-D Nicoud, and F. Mondada, editors, *Proceedings of the 5th European Conference on Advances in Artificial Life*, volume 1674, pages 504–513, 1998.
10. S. A. Frank. *Foundations of Social Evolution*. Princeton University Press, 1998.
11. S. Hirose, Y. Inazu, S. Chae, and Y. Maeda. Suppression of the growth/differentiation transition in *Dictyostelium* development by transient expression of a novel gene, dia1. *Development*, 127:3263–3270, 2000.
12. M. Kerszberg and L. Wolpert. The origin of metazoa and the egg: a role for cell death. *Journal of Theoretical Biology*, 193:535–537, 1998.
13. R. H. Kessin. *Dictyostelium: Evolution, Cell Biology, and the Development of Multicellularity*. Cambridge University Press, 2001.
14. A. F. M. Marée, A. V. Panfilov, and P. Hogeweg. Migration and thermotaxis of dictyostelium discoideum slugs, a model study. *Journal of Theoretical Biology*, 199:297–309, 1999.
15. J. A. R. Marshall and J. E. Rowe. Viscous populations and their support for reciprocal cooperation. *Artificial Life*, 9:327–334, 2003.
16. J. Maynard Smith and E. Szathmáry. *The Major Transitions in Evolution*. Oxford University Press, 1995.
17. R.E. Michod. Cooperation and conflict in the evolution of individuality. ii. conflict mediation. *Proceedings of the Royal Society of London: Series B: Biological Sciences*, 263:813–822, 1996.

18. H. Okada, Y. Hirota, R. Moriyama, Y. Saga, and K. Yanagisawa. Nuclear fusion in multinucleated giant cells during the sexual development of *Dictyostelium discoideum*. *Developmental Biology*, 118:95–102, 1986.
19. T. Pfeiffer and S. Bonhoeffer. An evolutionary scenario for the transition to undifferentiated multicellularity. *Proceedings of the National Academy of Sciences of the United States of America*, 100:1095–1098, 2003.
20. D. C. Queller. Relatedness and the fraternal major transitions. *Philosophical Transactions of the Royal Society of London. Series B.*, 355:1647–1655, 2000.
21. D.C. Queller. Genetic relatedness in viscous populations. *Evolutionary Ecology*, 8:70–73, 1994.
22. K. B. Raper. *The Dictyostelids*. Princeton University Press, Princeton, NJ, 1984.
23. John Stewart. Evolutionary transitions and artificial life. *Artificial Life*, 3:101–120, 1997.
24. M. Travisano and G.J. Velicer. Strategies of microbial cheater control. *Trends in Microbiology*, 12:72–78, 2004.
25. M. A. Wallace and K. B. Raper. Genetic exchanges in the macrocysts of *Dictyostelium discoideum*. *Journal of General Microbiology*, 113:327–337, 1979.
26. S.A. West, I. Pen, and Griffin A.S. Cooperation and competition between relatives. *Science*, 296:72–75, 2002.
27. L. Wolpert. The evolution of development. *Biological Journal of the Linnean Society*, 39:109–124, 1990.

Ant Clustering Embeded in Cellular Automata*

Xiaohua Xu[1], Ling Chen[2,3], and Ping He[2]

[1] Department of Computer Science and Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P.R. China
x.xu@citiz.net
[2] Department of Computer Science, Yangzhou University, Yangzhou 225009, P.R. China
{lchen@yzcn.net, angeletx}@citiz.net
[3] National Key Lab of Novel Software Tech, Nanjing University, Nanjing 210093, P.R. China

Abstract. Inspired by the emergent behaviors of ant colonies, we present a novel ant algorithm to tackle unsupervised data clustering problem. This algorithm integrates swarm intelligence and cellular automata, making the clustering procedure simple and fast. It also avoid ants' longtime idle moving, and show good separation of data classes in clustering visualization. We have applied the algorithm on the standard ant clustering benchmark and we get better results compared with the LF algorithm. Moreover, the experimental results on real world applications report that the algorithm is significantly more efficient than the previous approaches.

1 Introduction

Clustering is a very important problem in data mining. It classifies a mass of data, without any prior knowledge, to clusters which are clear in space partition outside and highly similar inside. Although many classic algorithms like k-mean have been proposed, still, they have their limitations. That is, in general, they need a great amount of prior-knowledge, e.g. the number of clusters or the initial data partition; however, the result is only local optimality can be achieved even with the partition principles.

Clustering is also shown as emergent behaviors of some social insect colonies in nature. In fact, such amazing complex behaviors have always been the focus of scientists, especially the ant colony. The ant colony can cluster through interaction and cooperation, and we call the clustering algorithm inspired by the ant behavior 'Ant Clustering Algorithm'.

Cellular Automata (CA), which was first presented by J.von Neumann [1], can be used in the research on information theory and algorithms. In CA, the concept of artificial ants was firstly proposed to represent cells. And recently, inspired by the swarm intelligence shown through the social insects' self-organizing behavior,

* This research was supported in part by Chinese National Science Foundation under contract 60473012 and Chinese National Foundation Science and Technology Development under contract 2003BA614A-14.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 562 – 571, 2005.

researchers created a new type of artificial ants to imitate the ants' behaviors in nature, and named it artificial ant colony system, another form of artificial life. Social insects have highly swarm intelligence [2], [3], such as reproducing, foraging, nest building, brood sorting and territory defending. Inspired by that, people have designed a series of algorithms that have been successfully applied to the areas of function optimization [3], combinational optimization [2], [4], network routing [5], robotics [6] and other scientific fields. Some researchers have achieved promising results in data mining by using the artificial ant colony. Deneubourg et al [7], [8] first proposed a Basic Model (BM) to explain the ants' behavior of piling corpses. Based on BM algorithm, Lumer and Faieta [9] presented a formula to measure the similarity between two data objects and designed the LF Algorithm for data clustering. BM and LF have become well-known models that have been extensively used in different applications. Kuntz et al [10] improved the LF algorithm and successfully applied it to graph partitioning and other related problems. Ramos et al [11] and Handl et al [12] recently applied the LF algorithm to text clustering and reported promising results.

However both BM and LF models separate ants from the to-be-clustered data objects and use much more parameters and information, resulting in the increase of the amount of the memory space and the data arrays to be processed. Moreover, since the clustered data objects cannot move automatically and directly, the data movements have to be implemented through the ants' movements. Considered that the ants would make idle movement when carrying no data object. it will inevitably lead to a large amount of extra information storage and computation burden. When the ant carrys an isolated data object, it may never find a proper location to drop it, and make an long-time idle moving. That will consume a large amount of computation time. If we are to form a high-quality clustering, the time cost is much higher.

According to Abraham Harold Maslow's hierarchy of needs theory [15], the security desire becomes more important for human beings when their primary physical needs are satisfied. By the same token, we notice that for such weak creatures as ants, they will naturally gather in groups with those who have similar features and repel those different. Even among a single group, there are separate nests, of which intimate ants build nests next to each other while strangers build nests far apart. We borrowed the principle of Cellular Automata in artificial life and proposed an Ants Sleeping Model (ASM) to explain the ants' behavior of searching for secure habitat. Based on ASM, we present an artificial ant algorithm of clustering (A^4C) in which each artificial ant is an intelligent agent representing an individual data object. We define a fitness function to measure the ants' similarity with their neighbors. Since each individual ant uses only a little local information to decide whether to activate or sleep, the whole group dynamically self-organizes into distinctive, independent subgroups which are highly similar inside.

The rest of the paper is organized as follows. Although the primer idea of ASM is firstly introduced in our paper [13] and [14], for the integrity of this article, section 2 presents ants sleeping model and its formal definition. Section 3 describes the implement of ant clustering algorithm based on ASM. Experimental results are shown in section 4 and section 5 concludes the paper.

2 Ants Sleeping Model

In the nature, ants tend to group with those that have similar features. Even within an ant group, they run to inhabit with familiar fellows as neighborhood. To satisfy their need for security, the ants are constantly choosing more comfortable and securer environment to sleep. Inspired by their behavior, we extend the classical CA model by combining it with the swarm intelligence, and present Ants Sleeping Model (ASM) for clustering problem in data mining.

ASM can be denoted by a 5-tuple ASM = (G, Q, N, δ, D). Here G represents a two-dimensional grid in which each position is a cell. Let $G = [0..w(n)\text{-}1]\times[0..h(n)\text{-}1]$ represent the cellular space, a 2-dimensional array of all position (x, y), where $x\in[0..w(n)\text{-}1]$, $y\in[0..h(n)\text{-}1]$, $n\in Z^+$, $h(n)\in Z^+$, $w(n)\in Z^+$, $w(n)$ and $h(n)$ are functions related to n, the number of *agent*. We set

$$w(n) = 2(\lfloor\sqrt{n}\rfloor+1),\ \ h(n) = 2(\lfloor\sqrt{n}\rfloor+1) \tag{2.1}$$

Each position on the grid is represented by a two-dimensional coordinate (x, y), and $G(x, y)\in Q$ represents the information at the position (x, y).

Q represents the set of cells' limited states. Let an *agent* represent one data object, and $agent_i$ represent the i^{th} *agent*. $N(agent_i)$ is $agent_i$'s neighborhood, and $N(x_i, y_i) = N(agent_i)$. We set

$$N(agent_i) = \left\{(x \bmod w(n), y \bmod h(n)) \;\middle|\; |x - x_i| \le s_x,\ |y - y_i| \le s_y \right\} \tag{2.2}$$

$$\partial N(agent_i) = \left\{(x_i \bmod w(n), y_i \bmod h(n)) \;\middle|\; |x - x_i| = s_x, |y - y_i| = s_y \right\} \tag{2.3}$$

where $s_x\in Z^+$, $s_y\in Z^+$. s_x and s_y are the vision limits of $agent_i$ in the horizontal and vertical direction. $\partial N(agent_i)$ is used to denote the bound of $N(agent_i)$. In ASM, each agent represents one data object. This pattern is closer to the clustering behavior in nature, because the agents (ants) behave just like birds of a feather flock together, rather than categorizing data objects in BM and LF.

We define δ as a set of the clustering rules which update the classes information of the agents. $\forall agent_i \in G,\ \delta(agent_i): Q^{|N(agent_i)|} \mapsto Q$. The next state of $agent_i$ is determined by δ through the interaction of $agent_i$ with its neighbouring cells. δ is also related to *agent's* activating probability p_a. The following rules must be included in δ:

- If $agent_i$ is sleeping, its class label is the same as most of its neighbors'.
- If $agent_i$ is active, its class label is the same as its label.

$D = (d_{ij})_{n\times n}$ represents the dissimilarity matrix among agents. Let $data_i = (z_1, z_2, \ldots, z_k)\in R^k$, $k\in Z^+$, we define

$$d_{ij} = d(agent_i, agent_j) = d(data_i, data_j) = \left\| data_i - data_j \right\|_p \tag{2.4}$$

$$D = (d_{ij})_{n\times n} = (d(agent_i, agent_j))_{n\times n} \tag{2.5}$$

Because each agent represents one data object, $d(agent_i, agent_j)$ is determined by the distance between the data objects represented by $agent_i$ and $agent_j$. Normally, we take Euclidean distance where $p = 2$.

We use $f(agent_i)$ to represent the current fitness of $agent_i$, in another word, how well does $agent_i$ fit into its current living environment.

$$f(agent_i) = \max\left\{0, \frac{1}{(2s_x+1)\times(2s_y+1)} \sum_{agent_j \in N(agent_i)} \left(1 - \frac{d(agent_i, agent_j)}{\alpha_i}\right)\right\} \tag{2.6}$$

$$\alpha_i = \frac{1}{n-1} \sum_{j=1}^{n} d(agent_i, agent_j) \tag{2.7}$$

$$\alpha = \frac{1}{n \times (n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} d(agent_i, agent_j) \tag{2.8}$$

Here α_i represents the average distance between $agent_i$ and other agents, and is used to determine the threshold value of its dissimilarity to other agents to leave them. Obviously, $f(agent_i) \in [0,1]$. Sometimes, we can use a constant α to substitute α_i, namely

$$f(agent_i) = \max\left\{0, \frac{1}{(2s_x+1)\times(2s_y+1)} \sum_{agent_j \in N(agent_i)} \left(1 - \frac{d(agent_i, agent_j)}{\alpha}\right)\right\} \tag{2.9}$$

We use function $p_a(agent_i)$ to denote the probability for $agent_i$ to be activated by the surrounding environment. We define

$$p_a(agent_i) = \frac{\beta^\lambda}{\beta^\lambda + f(agent_i)^\lambda} \tag{2.10}$$

Here $\beta \in R^+$ is the fitness threshold of $agent_i$' activation. Parameter $\lambda \in R^+$ is the activating pressure of the agents, normally we take $\lambda = 2$.

3 Implement of Ant Clustering Algorithm

In the initial stage of the algorithm, the agents are randomly scattered on the grid, no more than one agent per position. All of them are in active state, randomly moving around on the grid following the moving strategy. Here, the simplest moving strategy is to freely choose one unoccupied position in the neighborhood as the next destination. When an agent moves to a new position, it will recalculate its current fitness f and activating probability p_a so as to decide whether it should continue moving or sleep. If the current p_a is small, the agent has a lower probability of continuing moving and tends to take a rest at its current position. Otherwise the agent tends to keep in active state and continue moving. Here, the agent's fitness is related to its heterogeneity with its neighboring agents. When they are different from him, he feels insecure, and leaves his current position and searches for a more homogeneous position. When he finds it, the agent would stop moving then take a rest. During this course, the agents influence each other's fitness, but only limited to their neighboring agents. With increasing number of iterations, the agents with more homogeneity gather closer and those with more heterogeneity separate afar. Eventually, similar agents are gathered within a small area and have identical class label while different types of agents are located in separated areas and have different class labels. The framework of the algorithm is as follows.

Table 1. High-level description of A^{4}C (adaptive artificial ant clustering algorithm)

Algorithm A^{4}C
Input **:** the data object *data*[1 : *n*]
Output **:** the information of data clustering

```
01.  data preprocessing
02.  initialize grid and key parameters
03.  while (not termination) do
04.    for i ← 1 to n do
05.      agent_i.calculateActiveProbability( )
06.      r ← random([0, 1))
07.      if r ≤ p_a(agent_i) then
08.        agent_i.move( )
09.      else
10.        agent_i.sleep( )    // do nothing
11.      end if
12.      agent_i.update( )
13.    end for
14.    update key parameters adaptively
15.  end while
16.  for i ← 1 to n do
17.    output (i, agent_i.getClusterID)
18.  end for
19.  show grid G as an image
```

Some details of the algorithm are explained as follows.

```
Procedure initializeGrid
01.  perm ← randomPermutation([0..w(n)·h(n)-1])
02.  for i ← 1 to w(n)·h(n) do
03.    (x, y) ← (⌊perm(i)/h(n)⌋, perm(i) mod w(h))
04.    if i ≤ n do    // put down agent_i at (x, y)
05.      G(x, y) ← i
06.    else           // the location of (x, y) is empty
07.      G(x, y) ← 0
08.    end if
09.  end for
```

Procedure *calculateActiveProbability*

01. $(x_i,\ y_i) \leftarrow agent_i.getLocation(\ )$
02. $N(agent_i) \leftarrow \left\{(x_i \bmod w(n), y_i \bmod h(n))\middle|\ |x-x_i| \le s_x, |y-y_i| \le s_y\right\}$
03. $f(agent_i) \leftarrow \dfrac{1}{(2s_x+1)\times(2s_y+1)} \sum_{agent_j \in N(agent_i)} \dfrac{\alpha_i^2}{\alpha_i^2 + d(agent_i, agent_j)^2}$
04. $p_a(agent_i) \leftarrow \dfrac{\beta^\lambda}{\beta^\lambda + f(agent_i)^\lambda}$

Procedure *move*

01. $\partial N(agent_i) \leftarrow \left\{(x_i \bmod w(n), y_i \bmod h(n))\middle|\ |x-x_i| = s_x, |y-y_i| = s_y\right\}$
02. $L(agent_i) \leftarrow \left\{(x,\ y)\middle|\ (x,\ y) \in N(agent_i), G(x,\ y) = 0\right\}$
03. ***if*** *not* $isEmpty(L(agent_i))$ ***then***
04. $\quad (x,\ y) \leftarrow L(agent_i).getElement(\ )$
05. $\quad G(x_i,\ y_i) \leftrightarrow G(x,\ y)$
06. $\quad agent_i.setLocation(x,\ y)$
07. ***else***
08. $\quad (x,\ y) \leftarrow \partial N(agent_i).getElement(\ )$
09. $\quad j \leftarrow G(x,\ y)$
10. $\quad G(x_i,\ y_i) \leftrightarrow G(x,\ y)$
11. $\quad agent_i.setLocation(x, y)$
12. $\quad agent_j.setLocation(x_i, y_i)$
13. ***end if***
14. $agent_i.setSleepState(false)$

Procedure *sleep*

01. ***if*** *not* $agent_i.isSleep(\ )$ ***then***
02. $\quad agent_i.setSleepState(true)$
03. $\quad newClusterID \leftarrow N(agent_i).getMostClusterID(\ )$
04. $\quad agent_i.setClusterID(mostClusterID)$
05. ***end if***

Procedure *update*

01. $\alpha_i(t) = \alpha_i(t-\Delta t) - k_\alpha(\overline{f}_t - \overline{f}_{t-\Delta t})$
02. $agent_i.updateHistory(\ )$

The value of α is adjusted adaptively during the process of clustering. We use $\bar{f}_t$ to denote the average fitness of the agents in the t-th iteration. To a certain extent, $\bar{f}_t$ indicates the quality of the clustering. The value of α can be modified adaptively using formula (3.2), Here k_α is a constant.

$$\bar{f}_t = \frac{1}{n}\sum_{i=1}^{n} f_t(agent_i) \tag{3.1}$$

$$\alpha(t) = \alpha(t-\Delta t) - k_\alpha(\bar{f}_t - \bar{f}_{t-\Delta t}) \tag{3.2}$$

The active probability of each agent is computed using (2.10). In (2.10), we suggest $\beta \in [0.05,0.2]$ and normally $\beta = 0.1$. The parameter λ is a pressure coefficient of $p_a(agent)$. Normally, there are two methods to determine the parameter λ as follows: ① λ is a constant. We suggest $\lambda = 2$. ② Adjust the value of λ adaptively. Generally, the ants can form a rough clustering rudiment fast in the initial stage of clustering, while at the latter stage they require rather long time to improve the clustering because the precision needs to be raised. Therefore the activating pressure coefficient λ tends to change decreasingly on the whole. We adjust the value of λ adaptively with the following function.

$$\lambda(t) = 2 + \frac{k_\lambda}{\bar{f}_t}\lg\frac{t_{\max}}{t} \tag{3.3}$$

The activated $agent_i$ first select an empty position from its neighbors $L(agent_i)$ as its next position and then moves there. To determine $agent_i$'s next position, several methods can be used. ① The random method. $agent_i$ selects a location from $L(agent_i)$ randomly. ② The greedy method. Let a parameter $\theta \in [0,1]$ determine $agent_i$'s probability to select the most suitable location from $L(agent_i)$ as its destination. We denote it as θ-greedy selecting method. This method can make full use of the local information.

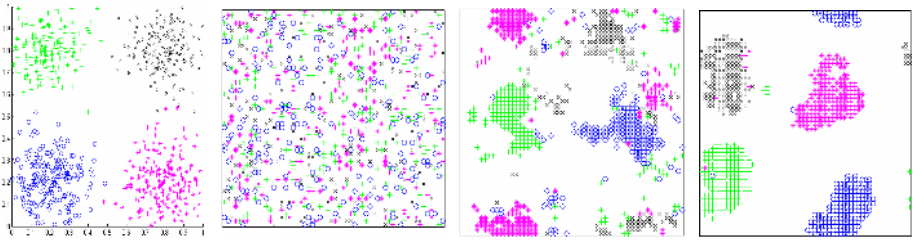
4 Experimental Results

In this section, we not only compared our test results on the ant-based clustering data benchmark, which was introduced by Lumer and Faieta in [9], with that of LF algorithm, but also show our test results on several real data sets.

4.1 Ant-Based Clustering Data Benchmark

First we test a data set composed of four data types, each of which consists of 200 two-dimensional data (x, y) as shown in Figure 1.(a). Here x and y obey normal distribution $N(\mu, \sigma^2)$. The normal distributions of the four types of data (x, y) are ($N(0.2,0.1^2)$, $N(0.2,0.1^2)$), ($N(0.2,0.1^2)$, $N(0.8,0.1^2)$), ($N(0.8,0.1^2)$, $N(0.2,0.1^2)$), and ($N(0.8,0.1^2)$, $N(0.8,0.1^2)$) respectively. Agents that belong to different cluster are represented by different symbols: o, +, *, ×.

We test the above benchmark using our algorithm A^{4}C. In the test the parameters are set as: β=0.1, s_x=1, s_y=1, θ=0.90, k_α=0.50 and k_λ=1. The initialized value of α is 0.5486. In each iteration, the value of α is updated adaptively by formula



(a) (b) (c) (d)

Fig. 1. The process of clustering of A^{4}C. (a) Data distribution in the attribute space; (b) Initial distribution of agents in the grid; (c) Agents' distribution after 10000 iterations in the grid; (d) Agents' distribution after 20000 iterations in the grid.

$\alpha(t)=\alpha(t-50)-k_\alpha(\bar{f}_t-\bar{f}_{t-50})$ and the agents select their destination using greedy method. Compared with the best results of the LF [9], our algorithm costs much less computation time (20000 iterations) than theirs (1000000 iterations).

4.2 Real Data Sets Benchmarks

We test LF and A^{4}C with the real benchmarks of Iris and Glass. In the test, we consider two types of A^{4}C: one sets the parameters adaptively and we still denote it as A^{4}C, the other uses constant parameters and we call it SA^{4}C (special A^{4}C). The clustering results of A^{4}C after 5000 iterations are mostly better than those of LF after 1000000 iterations. We set the parameter t_{max} the maximum iterations of LF algorithm as 1000000, and those for A^{4}C and SA^{4}C as 5000. And we set the parameters

Table 2. Parameters and test results of LF, SA^{4}C and A^{4}C on Iris

	LF	SA^{4}C	A^{4}C
Maximum iterations t_{max}	***1000000***	5000	**5000**
Numbers of trials	100	100	100
Minimum errors	3	2	**2**
Maximum errors	13	8	**5**
Average errors	***6.68***	4.39	**2.13**
Percentage of the errors	*4.45%*	2.94%	**1.31%**
Average running time(s)	***56.81***	**1.36**	1.43

Table 3. Parameters and test results of LF, SA^{4}C and A^{4}C on Glass

	LF	SA^{4}C	A^{4}C
Maximum iterations t_{max}	***1000000***	5000	**5000**
Numbers of trials	100	100	100
Minimum errors	7	4	**2**
Maximum errors	12	12	**10**
Average errors	***10.25***	7.59	**3.94**
Percentage of the errors	4.79%	3.55%	**1.84%**
Average running time(s)	***106.21***	**2.37**	2.44

k_1=0.10, k_2=0.15 in LF and θ=0.9, λ=2, k_α=0.5, k_λ=1, β=0.1 in SA^{4}C and A^{4}C. In all three algorithms, we set the size of the neighborhood as 3×3. Results of 100 trials of each algorithm on the two data sets are shown in Table 2 to 3. It can be easily seen from the tables that SA^{4}C and A^{4}C requires much less iterations than LF, while the quality of the clustering of SA^{4}C is still better than LF, however is worse than A^{4}C.

From the test results above, it is obvious that the time cost of LF is quite high, mainly because ants spend a great deal of time in searching for data and have difficulty to drop its data in the data-rich environment. Additionally, LF cannot deal with isolated points efficiently. Because it is very difficult for ants carrying an isolated data object to find a proper position to drop it down, they would possibly make longtime idle moving, which consumes large amount of computational time. With regard to the clustering quality, the parameters in LF algorithm, especially the parameter α, is difficult to set and can affect the quality of clustering results. Moreover, the parameters in LF lack adaptive adjustment which delays the process of clustering. Our A^{4}C algorithm based on ASM not only has advantages of simple, direct, dynamic and visible, but also offer self-adaptive adjustment to important parameters and can process isolated dada directly and effectively. Compared to LF algorithm, A^{4}C algorithm is more effective and can converge faster.

5 Conclusions

An artificial Ants Sleeping Model (ASM) is presented to resolve the clustering problem in data mining by simulating the emergent behaviors of ant colonies. In the ASM model, each data is represented by an agent. The agents' environment is a two-dimensional grid where each agent interacts with its neighbors and exerts influence on each other. Those with similar features form into groups, and those with different features repel. In addition, we proposed effective formulae to compute the fitness and activating probability of agents based on ASM model. We also present an Adaptive Artificial Ants Clustering Algorithm (A^{4}C). In A^{4}C, the agents can form into high-quality clusters by making simple movements according to little local neighborhood information, while the parameters are selected and adjusted adaptively. It has fewer restrictions on parameters, requires less computational cost, and shows better clustering quality. Experimental results on standard ant clustering benchmarks demonstrate that ASM and A^{4}C are more direct, easier to implement, and more efficient than BM and LF. The results of the test on the real world datasets undoubtedly confirm us that our algorithm is more efficient than the previous approaches.

References

1. von Neumann J.: Theory of self reproducing cellular automata. University of Illinois Press, Urbana and London (1966)
2. Bonabeau E., Dorigo M., Théraulaz G.: Swarm Intelligence: From Natural to Artificial Systems. Santa Fe Institute in the Sciences of the Complexity, Oxford University Press, New York, Oxford (1999)

3. Kennedy J., Eberhart R.C.: Swarm Intelligence. San Francisco, CA: Morgan Kaufmann Publishers (2001)
4. Dorigo M., Maniezzo V., Colorni A.: Ant system: optimization by a colony of cooperative learning approach to the traveling agents. IEEE Trans. On Systems, Man, and Cybernetics, 26(1) (1996) 29–41
5. Di Caro G., Dorigo M.: AntNet: A mobile agents approach for adaptive routing. Technical Report, IRIDIA (1997) 97–12
6. Holland O.E., Melhuish C.: Stigmergy, self-organization, and sorting in collective robotics. Artificial Life 5 (1999) 173–202.
7. Dorigo M., Bonabeau E., Théraulaz G.: Ant Algorithms and stigmergy. Future Generation Computer Systems 16 (2000) 851–871.
8. Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks A., Detrain, C., Chretien: L. "The Dynamic of Collective Sorting Robot-like Ants and Ant-like Robots". SAB'90 - 1st Conf. On Simulation of Adaptive Behavior: From Animals to Animats, J.A. Meyer and S.W. Wilson (Eds.). MIT Press (1991) 356–365
9. Lumer E., Faieta B.: Diversity and adaptation in populations of clustering ants. in: J.-A.Meyer, S.W. Wilson(Eds.), Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animats, Vol. 3, MIT Press/Bradford Books
10. Kuntz P., Layzell P., Snyder D.: A colony of ant-like agents for partitioning in VLSI technology. in: P. Husbands, I. Harvey(Eds.), Proceedings of the Fourth European Conference on Artificial Life. MIT Press, Cambridge, MA (1997) 412–424.
11. Ramos, Vitorino and Merelo, Juan J: "Self-Organized Stigmergic Document Maps: Environment as a Mechanism for Context Learning", in E.Alba, F. Herrera, J.J Merelo et al.(Eds.), AEB'02 – 1st Int. Conf. On Metaheuristics, Evolutionary and Bio-Inspired Algorithms. Mérida, Spain (2002) 284–293
12. Handl, J. and Meyer: B. Improved ant-based clustering and sorting in a document retrieval interface. PPSN VII, LNCS 2439 (2002)
13. Chen L., Xu X., and Chen Y.: An Adaptive Ant Colony Clustering Algorithm. Proc. Third International Conference on Machine Learning and Cybernetics (ICMLC'04) (2004) 1387–1392.
14. Chen L., Xu X., Chen Y., and He P.: A Novel Ant Clustering Algorithm Based on Cellular Automata. Proc. IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04) (2004)
15. http://www.maslow.org/

Ant-Based Computing

Loizos Michael

Division of Engineering and Applied Sciences,
Harvard University, Cambridge, MA 02138, U.S.A
loizos@eecs.harvard.edu

Abstract. We propose a biologically and physically plausible model for ants and pheromones, and show this model to be sufficiently powerful to simulate the computation of arbitrary logic circuits. We thus establish that coherent deterministic and centralized computation can *emerge* from the *collective* behavior of simple distributed markovian processes as those followed by ants.

1 Introduction

Entomological studies suggest that ants interact with each other and their environment in very specific, usually reactive ways, without individually performing complicated computations, and always in a probabilistic and distributed manner. Such kind of behavior seems to be necessary, and perhaps optimal, for guaranteeing survival of ants in the uncertain natural environment. In this work we ask what else (besides "mere" survival) is such a behavior good for, what the computational power of ant-like protocols is, and how can the nature of problems that are (or are not) solvable by such protocols be characterized.

Viewed as abstract processes, ants distributively execute simple memoryless probabilistic algorithms. Each process is markovian, and evolves as a function of the current state of the world only. Communication between processes is very limited and comes indirectly through the interaction of these processes with their environment. In this work we establish that such ant-like processes, coupled with appropriately defined initial conditions, are sufficient to simulate the computation of logical circuits one finds in modern digital computers. This is a rather surprising result, given that the very nature of these processes is in direct contrast to the design principles followed by digital computer hardware, namely binary logic, deterministic computation, centralized and complex decisions.

We conclude that ants and ant-like processes can *collectively* compute much more than the sum of the individual parts; they can perform arbitrary deterministic computations, and can store and retrieve state. Thus, the markovian behavior of the individual processes is surpassed by the collective workings of a number of processes. In the collective setting the processes are no longer the computational units, but are simply carriers of information. The carriers exhibit limited intelligence, and the actual computation *emerges* as a result of the interaction of these carriers through their environment.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 572–583, 2005.

1.1 Related Work

Our endeavor is not the first approach to understanding what the computational capability of ants is. Previous work has shown that ants are capable of solving non-trivial problems such as finding shortest paths between two points [3], and sorting multiple items into piles [4]. Our goal, however, differs significantly in that we attempt to investigate whether ants are capable of solving a problem, namely circuit computation, that ants do not (to the best of our knowledge) undertake in their natural environment. This is unlike the problems of finding a shortest path to a food source or to the nest, and of sorting the brood, the food, and the dead ants into different piles, which seem to be essential tasks for the survival of ants.

Related fields in Computer Science that have enjoyed a lot of attention over the past decade are that of Ant-Based Clustering [7] and Ant Colony Optimization [5]. Algorithms inspired by the behavior of ants are employed to heuristically cluster data according to some similarity measure, or to heuristically solve combinatorial optimization problems that are currently thought to be intractable. Despite being inspired by ant behavior, algorithms in these fields often employ state-based ant-agents whose behavior depends on the actual problem being solved. In addition, ant-agents only perform part of the actual computation, with a centralized entity actually monitoring and affecting their behavior over time. In contrast, we are concerned with developing a biologically plausible ant-based model of computation, where ants are memoryless and oblivious to the problem under consideration. The solution to the problem emerges only from the collective behavior of ants, and the computation is truly distributed without any centralized control.

Constructing circuits using biological or physical substrates has also been explored in the past. Cells have been manipulated to compute simple logical circuits [6], and fluids have been shown capable of computing the basic logic gate operations [8]. The problem we investigate here complements the existing approaches, in that it provides yet another biologically-inspired domain for creating circuits. We believe that ants are, in fact, a more appropriate metaphor for current flowing through electronic circuits, rather than different kinds of proteins used in cell-computing and different colors of fluids used in fluid computing, since unlike the latter approaches, our approach treats all "electrons" equivalently. This makes our approach modular and circumvents the limitations of other non-modular attempts that can (at the time of this writing) only scale up to the design and implementation of single gates or very small circuits.

On the other hand, one could argue that it is unlikely that a solution to the ant-based computing problem will prove of any practical importance in real life, unlike the other two approaches that seem to have real world applications. Although we agree that ant-based computers will probably not appear in households any time soon (except perhaps as a novelty item), we feel that the study we perform on how simple probabilistic distributed processes, as the ones employed by ants, can be used to compute logical circuits that exhibit a complex

coherent global behavior, is of independent interest in other research areas dealing with the study of emergent coherent behavior from unreliable components (e.g., amorphous computing [1]).

2 Modeling Ants and Pheromones

The two key building blocks of an ant-based computer are ants and pheromones. In this section we propose a model on their behavior and state any assumptions we make for the rest of the paper.

Ants are viewed at an abstract level as processes that operate on their environment on a discrete time basis. At each time step, an ant senses the pheromone concentrations at its current location and the locations reachable (based on the ant's position, direction, and physical constraints) therefrom. Based on this limited sensory input, a very simple algorithm computes the ant's action, which includes choosing whether pheromone should be secreted at the current location, choosing which direction to follow, and moving one distance unit towards the chosen direction. The computation taking place is in its nature reactive. An ant does not keep track of its past sensory input, and can only thus base its decision on the current state of its environment. The decision is taken probabilistically, in the sense that an ant reaching a given state twice, will not necessarily make the same choices; nonetheless, the probability distribution over the possible choices in a given state is fixed across the entire history of an ant (see e.g., [3]). In addition, the distribution is identical across all ants, but is however, completely independent between ants, giving the ants as a whole a distributed behavior.

We employ a single type of recruiting pheromone in this work. Ants are attracted to the pheromone, and in particular are more likely to move towards a position that has a higher, rather than a lower, concentration of the pheromone. Ants are also able to determine whether a sensed pheromone concentration is sufficiently above or sufficiently below some threshold that is fixed across ants (see [2] for a somewhat different approach). An ant that senses a sufficiently high pheromone concentration at its current location, and a sufficiently low pheromone concentration at some reachable location, will secrete pheromone at its current location (thus indirectly increasing the pheromone concentration at the adjacent locations and helping in recruiting other ants). The Algorithm ChooseAction$(\cdot,\cdot)$ describes formally the behavior of each individual ant at each time step.

The constant T in the algorithm represents the threshold on which ants base their decision on secreting pheromone, while ε stands for the gap that accounts for the ants' imperfect sensing. Finally, n stands for the degree of non-linearity in the probabilistic behavior of ants; the higher the n is, the more drastically the ants change their behavior when faced with a small difference between two pheromone concentrations.[1]

[1] In case all pheromone concentrations are zero, the new location is chosen uniformly at random amongst the reachable locations.

ChooseAction(Current Location L_C, Current Direction D_C)
1: Identify the set $\mathcal{R}(L_C, D_C)$ of all locations reachable in one step.
2: Sense pheromone concentration $\mathcal{P}(L)$ for each $L \in \{L_C\} \cup \mathcal{R}(L_C, D_C)$.
3: If $\mathcal{P}(L_C) \geq T + \varepsilon$ and there exists $L \in \mathcal{R}(L_C, D_C)$ s.t. $\mathcal{P}(L) \leq T - \varepsilon$, then secrete a fixed amount of pheromone at L_C.
4: Choose $L_N \in \mathcal{R}(L_C, D_C)$ with probability $\mathcal{P}(L_N)^n / \sum_{L \in \mathcal{R}(L_C, D_C)} \mathcal{P}(L)^n$.
5: Move to location L_N with direction D_N defined by vector $\overline{L_C L_N}$.

Algorithm 1. The markovian decision-making process of each individual ant

In addition to being secreted by ants, a certain amount of pheromone is pumped into the system at specific locations, after the lapse of every time unit. Each pump releases pheromone at a constant rate, although the rate might differ across pumps. A constant fraction of the pheromone from each location dissipates into the environment at every time step. What is more, pheromone diffuses across adjacent locations, by flowing from higher concentrations towards lower concentrations, moving closer to a uniform distribution across the system. The change of pheromone concentrations over time is formally described by the equation below.

$$\mathcal{P}^t(L) = (1-d) \cdot \left[(1-f) \cdot \mathcal{P}^{t-1}(L) + f \cdot \mathcal{W}^{t-1}(L)\right] + s^{t-1}(L) + p(L)$$

Time-dependent functions are superscripted with the time step. The function $\mathcal{P}^t(\cdot)$ maps a location to the amount of pheromone present at the location, while the function $\mathcal{W}^t(\cdot)$ maps a location to the average pheromone concentration of that location and all its adjacent locations. The function $s^t(\cdot)$ maps a location to the amount of pheromone secreted by ants at the location according to the protocol we have described above. The function $p(\cdot)$ maps a location to the (time-independent) amount of pheromone pumped into that location during any given time step. The constant d determines the percentage of the concentration of a pheromone that dissipates into the environment, while the constant f determines how uniformly the pheromone will diffuse during any given time step. Initially, all locations have a zero pheromone concentration.

2.1 Biological and Physical Plausibility

We argue that our proposed model for ants and pheromones is a plausible one. Our model assumptions on (i) the attraction of ants towards higher concentrations of recruiting pheromone, (ii) the ability of ants to sense pheromones and change their behavior accordingly, and (iii) the secretion of additional pheromone to attract more ants, are based on empirical evidence on the behavior of real ants. Regarding the behavior of pheromones, our model relies on two physical principles: (i) that gas dissipates to the environment at a rate that is proportional to its concentration, and (ii) that gas concentrations tend to become locally (and eventually globally) uniform with the lapse of time at a

rate that is proportional to the local gradient of the current concentrations. In fact, we argue that our model assumptions are in some sense necessary, or minimal, in order to achieve the desired goal of building an ant-based computer.

The bias that ants exhibit towards higher pheromone concentrations is essential in that it provides a minimal requirement for an otherwise probabilistic process to behave in some predictable manner. We extract some sort of deterministic behavior from ants by recognizing that the statement that "ants will more likely follow higher pheromone concentrations" is a statement that is always (i.e., deterministically and not probabilistically) true, albeit it, in itself, refers to a probabilistic process.

The secretion of recruiting pheromone by ants in the presence of increased (i.e., sufficiently above the threshold) pheromone concentrations is also essential, since it corresponds to a way for ants to access a shared memory and thus communicate. Notice that despite the minimalistic nature of the communication, some sort of messaging is nonetheless important in order to go from the distributed and independent behavior of individual ants, to the coherent and globalized computation needed for an ant-based computer.

Overall, our proposed model and assumptions can be viewed under a more general prism and abstracted away from ants and pheromones. One need only consider probabilistic processes and simple messages exchanged between them. The environment in which these processes reside is such that it constrains the processes from taking arbitrary actions, and it rewards (but not forces) processes for following certain rules (in some sense one could say that ants move towards higher pheromone concentrations because they find it rewarding). Thus, our model and results later on are applicable to a much more general setting and illustrate that robust global behavior can be obtained from probabilistic distributed processes.

3 Building an Inverter

The single most important component in a logical circuit is the inverter. An inverter takes a single truth value as its input and maps it to its logical negation as its output. In this section we propose a construction of an inverter based on ants and pheromones and the model thereof proposed in the previous section.

3.1 Engineering Analysis

An inverter is a set of positions, represented as hexagonal cells in Figure 1. Dark cells represent walls that cannot be crossed by ants, and through which the pheromone does not diffuse. Grey cells represent the paths that ants can follow and through which the pheromone diffuses; the assumption is that these paths are narrow enough so as ants cannot change their direction, unless they encounter a choice point (see e.g., [3]). Cells marked with a plus sign represent sources of ants, where new ants appear at every time step, while cells marked with a minus

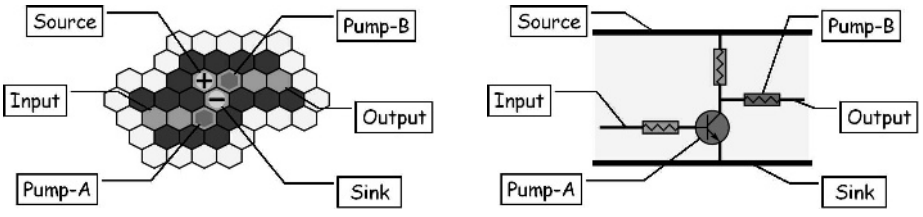


Fig. 1. An ant-based inverter and an electronic RTL inverter

sign represent sinks of ants, where ants that reach those cells disappear. Cells marked with a dark hexagon represent pheromone pumps; the rate at which pheromone is pumped into the system is implementation-depended and might differ across pumps. Finally, white cells represent the surrounding area of the inverter.

Ants enter the inverter at point *Input* and move towards point *Pump-A* and then towards point *Sink*, where they leave the system. New ants enter the system and the inverter at point *Source*, where they are faced with a choice point. Ants either choose to move towards point *Sink*, where they leave the system, or choose to move towards point *Pump-B* and then towards point *Output*, where they exit the inverter.

We note here that although the output of the inverter is driven by the input, the ants reaching the output do not come from the input, but rather from the source. The implications of this design and implementation choice are twofold. First, any specific ant only traverses a small distance, since when it enters the next inverter it will leave the system through the inverter's sink. Second, even if an ant starts moving backwards in a path, it will eventually leave the system through some preceding inverter's sink. Both implications support the claim that our implementation can scale up to large circuits, since ants (both well-functioning and ill-functioning ones) only traverse a distance that does not increase with the complexity of the circuit.

It worths pointing out the extreme resemblance of the various parts of an ant-based inverter to the parts of an electronic RTL inverter. In both inverters the input ends up reaching the sink/ground. When ants/current are/is present at the input, the input attracts/drives the source ants/current towards the sink/ground, by increasing/decreasing the pheromone/resistance compared to the other choice of moving towards the output.[2]

3.2 Theoretical Analysis

Any implementation of an inverter should respect a necessary set of design principles that make the inverter functional. In particular, the output, in addition to depending inversely on the input, should exhibit sufficient indifference to its

[2] The similarities between ant-based circuits and electronic circuits extend to a much deeper level than what is presented here. However, due to space limitations we defer such a discussion to an extended version of this paper.

input so as to be able to be driven by the merged output of at least two other inverters, and should exhibit sufficient output gain so as to be able to drive at least two other inverters.

In our model, we define what constitutes a logical 1 and what constitutes a logical 0 for our inverter by taking the average flow of ants through a given location over a fixed period of time. This average flow takes a value in the interval $[0, 1]$. The inverting behavior corresponds to the following: "When the input flow is sufficiently close to 0, the output flow is sufficiently close to 1, and when the input flow is sufficiently close to 1, the output flow is sufficiently close to 0". We, therefore, define a logical 0 to correspond to a flow in the interval $[0, x]$, and define a logical 1 to correspond to a flow in the interval $[y, 1]$. Of course, these two intervals should not overlap, and in fact they should be sufficiently apart; we thus require that $x \ll y$.

In order to account for sufficient input indifference and output gain, it suffices for the output of an inverter to produce amplified signals with respect to the ones defined above. We define an *amplified logical* 0 to correspond to a flow in the interval $[0, x/2]$, and define an *amplified logical* 1 to correspond to a flow in the interval $[2y, 1]$. It is, then, easy to see that merging or splitting two amplified signals results in a signal that can still be properly recognized by an inverter as a logical 0 or a logical 1.

The intervals $[0, x]$ and $[y, 1]$ described above are known as the *noise regions*, or the *noise immunity levels* of an inverter. They correspond to the intervals within which the signal is allowed to fluctuate due to external noise, without affecting the inverter's logical state. In our case, this "noise" corresponds to the variance of ant behavior with respect to their expected behavior, but it also encompasses the effects of splitting and merging ant flows in a circuit.

3.3 Experimental Analysis

For our experiments we have analyzed the behavior of a single inverter in response to different input values. At each time step, we have recorded the state of the inverter, comprised of the presence of ants and pheromone concentrations at each location. We evolved the state according to the model described in Section 2, by empirically setting the model parameters to appropriate values, as shown in Table 1.

We note that in our simulation we have relaxed the assumption that ants appear in the inverter's source at each time step, to the assumption that this happens with high probability. The results show that this relaxation does not impact the proper working of the inverter, and indicate that relaxing other assumptions (e.g., that all ants move at every time step) is possible, without unwanted consequences.

The graphs in Figure 2 show that as the input ant flow increases, the pheromone concentration at point *Sink* increases and exceeds the pheromone concentration at point *Pump-B*, making thus the ants move towards the inverter's sink and reduce the ant flow to the inverter's output. The effect of the change in pheromones is shown in Figure 3 depicting the ant flow at the

Table 1. Parameters and associated values used in our experimental setting

Description of model parameters	Empirical values
Probability of new ant appearing at source	95% per time unit
Exponent n in ant probability function	30
Threshold T for pheromone concentration	6 units
Threshold accuracy error ε	0.1 units
Pheromone amount secreted by ants	12 units per secretion
Pheromone dissipation rate d	10% per time unit
Pheromone diffusion rate f	10% per time unit
Pheromone pumped in at point *Pump-A*	1 unit per time unit
Pheromone pumped in at point *Pump-B*	0.2 units per time unit

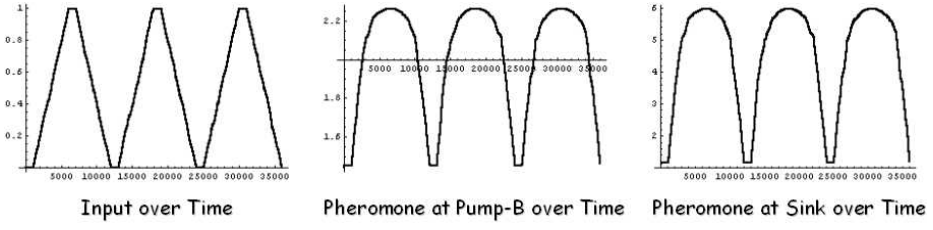


Fig. 2. The pheromone concentrations at the choice point of an inverter

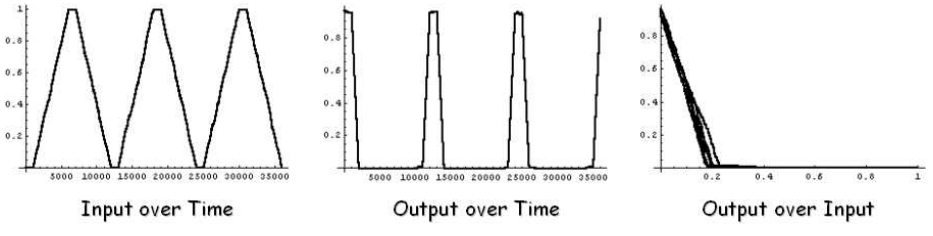


Fig. 3. The ant flows at the input and output locations of an inverter

inverter's output changing inversely to the input ant flow. The last of the three graphs shows the amplification performed by the inverter.

The width of this last graph corresponds to the variance in the behavior of the inverter with respect to its expected behavior. Even by taking the least amplified output value at every input value (i.e., the highest output value when considering a logical 1 input, and the lowest output value when considering a logical 0 input), one can see that setting $x = 0.05$ and $y = 0.22$ leads to noise regions that more than satisfy the conditions set forth in Section 3.2. In particular, when the input lies in the interval $[0, x]$ the output lies in the interval $[3y, 1]$ and when the input lies in the interval $[y, 1]$ the output lies in the interval $[0, x/5]$; again, these regions take into account the worst observed behavior in

our experiments. The fact that these intervals exhibit an output gain factor of 3 and an input indifference factor of 5, gives our implementation additional noise robustness with respect to the theoretical requirement for these factors to be at least 2.

4 Gates, Circuits, Computers

In addition to an inverter, one needs to define a set of other primitive components, which can then be combined into logical gates, circuits, and ant-based computers.

4.1 Primitive Components

Figure 4 shows the complete list of primitive components, which we describe briefly below. A wire is simply a path surrounded by walls that guide ants in a

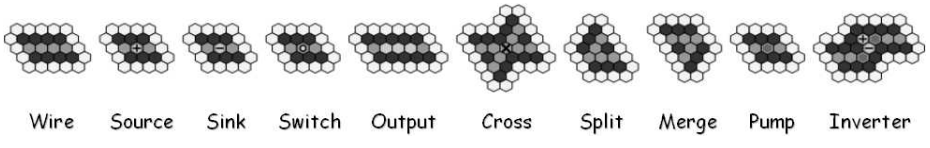


Fig. 4. List of circuit components

given direction. The positive battery side corresponds to an ant source, while the negative battery side to an ant sink. A switch is simply a controlled ant source where some sort of valve is used to start or stop the flow of ants. The output of a circuit is simply a designated part of some wire that one can observe; the rest of the circuit need not be directly observable. A wire-cross is a bridge that allows one path to pass on top of another.[3] A wire-split is a choice point for ants (the only other choice point is the one in the inverter), where a path splits into two, while a wire-merge is when two paths merge into one. By appropriately choosing the width of the paths at the merging point, one can physically restrict incoming ants to only choose the outgoing path. Finally, a pump is a device that releases pheromone into the system at a constant rate.

4.2 Basic Logic Gates

One can combine primitive components to build the basic logic gates, as illustrated in Figure 5. From a theoretical point of view, we know that the set of

[3] It is known that every circuit is planar, meaning that one can always draw an equivalent circuit (i.e., a circuit that computes the same function) that does not contain any wire-crossings. However, this new circuit is sufficiently more complex and less natural than the original circuit, that justifies the use of wire-crossings for simplicity reasons.

operators $\{\neg, \vee\}$ is sufficient to express any binary operator. In our case the inverter plays the role of the negation operator, while the wire-merge (also known as wire-or in circuit design) plays the role of the disjunction operator. In order to normalize the merged flow of ants that come out of a wire-merge, we always follow the wire-or by an inverter, or an amplifier. By appropriately negating the inputs and output of a wire-or we get the basic logic gates. Since our inverter is amplifying its output, it follows that all the gates are by construction amplifying their output.

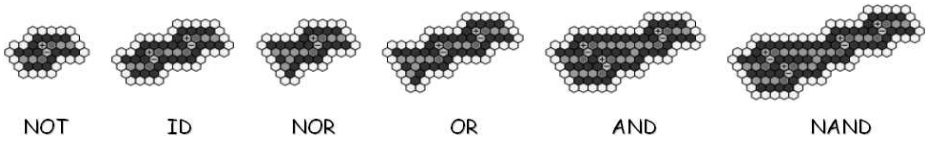


Fig. 5. List of the basic logic gates

4.3 Circuits and Computers

Using the gates and the components we can now build circuits. Figure 6 shows how one can build some basic circuits that can be found in a computer's RAM (Random Access Memory) and ALU (Arithmetic and Logic Unit).

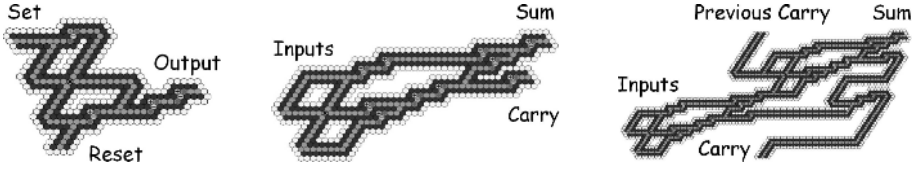


Fig. 6. A one-bit memory, a half-adder, and part of a multi-bit adder

Of course one can keep building more and more elaborate circuits up to an ant-based computer. We do not, however, pursue the design of more complex circuits in this work, since it does not offer any additional insight into our primary goal which is concerned with establishing the feasibility of such a task; we feel that the circuits we have presented suffice to establish such a claim.

4.4 Battery Implementation

We finally address the issue of how ants appear in inverters' sources and disappear in inverters' sinks. Figure 7 illustrates the implementation of an ant-based battery and its parallels to an electronic battery.

The battery implementation relies on extending the circuit level with two additional levels above and below the circuit. The three levels connect with each other through funnels; ants falling into a funnel end up one level below the one they started at. Initially, only the source level contains ants. Ants fall through

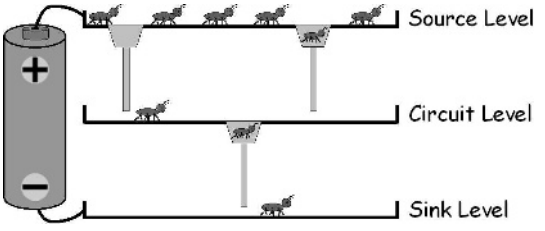


Fig. 7. An electronic battery and an ant-based battery

the top funnels into the circuit level, at the inverters' source locations. Ants then move around in the circuit level until they reach the inverters' sink locations, at which point they fall through the bottom funnels into the sink level, where they stay from that point onwards. The battery empties up when the source level no longer contains enough ants to support a steady flow of ants into the circuit. By manually moving the ants from the sink level to the source level one effectively recharges the battery.

5 Conclusions and Future Work

We have shown that markovian processes, as those followed by ants, can collectively produce a global coherent behavior. We have illustrated this by proposing a simple and intuitive model on the behavior of ants and pheromones and showing that our model is sufficiently rich to allow ants to compute arbitrary logic circuits, as the ones found in digital computers. We have argued and shown through experiments that our model possesses two properties that are individually easy to obtain, but seem to be hard to accommodate simultaneously: the model is biologically and physically plausible, and at the same time it is expressive enough to make the construction of circuits possible.

As future work one can pursue both further theoretical analysis, additional experimentation, and actual implementation of an ant-based computer. The most likely candidate for theoretical analysis is to solve the set of equations that describe the change in the state of an inverter, and prove what we have empirically observed, namely that these equations have a steady state for any given input to the inverter. In addition, one might try to compute the convergence time until a steady state is reached (i.e., the response time of the inverter). This can also be empirically measured and cross-checked with the theoretical bounds. Finally, one can prove bounds on the probability that the inverter will work correctly. Since each ant acts independently, there is a non-zero probability that a large number of ants will not behave as expected, and that the inverter will produce a bogus output. However, exactly because of the independence between ants, the probability of this happening is extremely small. Although our existing empirical results already support this claim, it would be useful to also have an analogous theoretical result.

On the experimental side, one could explore other sets of parameters that make the inverter work, perhaps guided by theoretical results on what the relation between these parameters should be. Preliminary experimental results towards this direction have indicated that our model is robust to changes of parameters, as long as the parameters are changed in meaningful ways (e.g., when increasing the dissipation rate, one would probably need to increase the amount of pheromone entering the system as well). A second experimental direction would be to simulate an entire circuit (perhaps a ring-oscillator, a multi-bit adder, or a memory unit) and show that the circuit works as expected. A simulation that displays the circuit and the ants moving would provide an extremely interesting (and fun) visualization of the circuit computation. The final frontier in verifying our model and circuit design would be, of course, to build circuits with actual ants. Such a study lies perhaps outside the Computer Science scope of this work and is, therefore, left to entomologists!

Acknowledgments. The author would like to thank his father, George Michael, for many hours of useful discussions on various aspects of this work, and especially for his help in clarifying the finer points in the parallels between ant-based circuits and electronic circuits.

References

1. H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. F. Knight, R. Nagpal, E. Rauch, G. J. Sussman, and R. Weiss: Amorphous Computing. In *Communications of the Association for Computing Machinery* 43(5), 2000.
2. E. Bonabeau, G. Theraulaz, and J.-L. Deneubourg: Fixed Response Thresholds and the Regulation of Division of Labour in Insect Societies. In *Bulletin of Mathematical Biology*, 60:753-807, 1998.
3. J.-L. Deneubourg, S. Aron, S. Goss, and J. Pasteels: The Self-Organising Exploratory Pattern of the Argentine Ant. In *Journal of Insect Behavior* 3:159-168, 1990.
4. J.-L. Deneubourg, S. Goss, N. Franks, A.Sendova-Franks, C. Detrain, and L. Chrétien: The Dynamics of Collective Sorting: Robot-Like Ants and Ant-Like Robots. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, 1991.
5. M. Dorigo, and T. Stützle: *Ant Colony Optimization*, MIT Press, U.S.A., 2004.
6. T. F. Knight, and G. J. Sussman: Cellular Gate Technology. In *Proceedings of the First International Conference on Unconventional Models of Computation*, 1998.
7. E. Lumer, and B. Faieta: Diversity and Adaption in Populations of Clustering Ants. In *Proceedings of the Third International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*, 1994.
8. T. Vestad, D.W.M. Marr, and T. Munakata: Flow Resistance for Microfluidic Logic Operations. In *Applied Physics Letters* 84:5074-5075, 2004.

Collective Behavior Analysis of a Class of Social Foraging Swarms*

Bo Liu, Tianguang Chu, and Long Wang

Intelligent Control Laboratory, Center for Systems and Control,
Engineering Research Institute, Peking University, Beijing 100871, P. R. China
{boliu, chutg, longwang}@pku.edu.cn

Abstract. This paper considers an anisotropic swarm model that consists of a group of mobile autonomous agents with an attraction-repulsion function that can guarantee collision avoidance between agents and a Gaussian-type attractant/repellent nutrient profile. The swarm behavior is a result of a balance between inter-individual interplays as well as the interplays of the swarm individuals (agents) with their environment. It is proved that the members of a reciprocal swarm will aggregate and eventually form a cohesive cluster of finite size. It is shown that the swarm system is completely stable, that is, every solution converges to the equilibrium point set of the system. Moreover, it is also shown that all the swarm individuals will converge to more favorable areas of the Gaussian profile under certain conditions. The results of this paper provide further insight into the effect of the interaction pattern on self-organized motion for a Gaussian-type attractant/repellent nutrient profile in a swarm system.

1 Introduction

Social behavior in swarms of entities is ubiquitous in nature. For example, bees, ants and birds often work together in groups for viability [1]. It is known that such cooperative behavior has certain advantages such as increasing the chance of finding food and avoiding predators and other risks. Understanding the cooperative and operational principles of such systems may provide useful ideas for developing formation control of multi-robots, cooperative control for uninhabited autonomous air/sea vehicles, and intelligent vehicle highway systems [2]–[3]. This has motivated an increasing interest in modeling and exploiting collective dynamics of swarms in ecology, biology, physics and more recently, in engineering applications (see, e.g. [4]–[10] and the references therein).

The general understanding in biology is that the swarming behavior is a result of an interplay between a long range attraction and a short repulsion between the individuals [1]. Recently, Gazi *et al.* [7] proposed an isotropic swarm model

* This work was supported by NSFC (60274001, 10372002) and National Key Basic Research and Development Program (2002CB312200). Corresponding author: T. Chu. E-mail: chutg@pku.edu.cn.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 584–593, 2005.

with a simple attraction-repulsion function governing the inter-individual interactions and showed that the model can exhibit the basic features of aggregation, cohesion and complete stability. But they only considered an all-identical coupling pattern in the swarm model. However, in many cases interaction strength between individuals in a swarm system may vary from one pair to another, depending on relative locations of the individuals or other factors, such as different communication efficiency and bandwidth. More recently, Chu *et al.* [8] studied anisotropic swarm models and found that the swarm dynamics relies crucially on the coupling patterns of the swarms. Moreover, in the course of foraging the swarm individuals may also be affected by a nutrient profile (or an attractant/repellent), that is, attraction to the more favorable areas or repulsion from the unfavorable areas of the attractant/repellent profile. Gazi and Passino [9] considered this case and proposed a simple model that can capture basic features of aggregation, cohesion and stability of social foraging swarms. However, the attraction-repulsion functions considered in these studies could not avoid collisions since they are not unbounded for infinitesimally small arguments. Therefore, it is still of interest to investigate more general anisotropic swarms.

In this paper we present a new anisotropic swarm model with a Gaussian-type attractant/repellent profile and an attraction-repulsion function that can avoid collisions and analyze the swarm dynamics under the interplays of the swarm individuals and the affect of environment. Specifically, we analyze stability of the swarm and show that the individuals will form a cohesive swarm of finite size and will converge to more favorable areas of the profile. The studies here further extend our recent work [10] to more general case.

This paper is organized as follows. Section 2 presents the swarm model. Section 3 analyzes the swarm behavior of aggregation, cohesion and complete stability under quasi-reciprocal conditions on coupling patterns of the swarm. Section 4 gives numerical simulations to demonstrate complex self-organized oscillation in general nonreciprocal swarms. The paper is conclude in section 5.

2 The Swarm Model

We consider a swarm of M individuals (or members) in an n-dimensional Euclidian space whose motion is governed by the following differential equations

$$\dot{x}^i = -\nabla_{x^i}\sigma(x^i) + \sum_{j=1, j\neq i}^{M} w_{ij} g(x^i - x^j), \quad i = 1, 2\cdots, M \tag{1}$$

where $x^i \in \mathbb{R}^n$ represents the position of individual i, $W = [w_{ij}] \in \mathbb{R}^{n\times n}$ is the coupling weight matrix with $w_{ij} \geq 0$ and $w_{ii} = 0$ for all i, j, $\sigma : \mathbb{R}^n \to \mathbb{R}$ represents the attractant/repellent profile or the "σ-profile" which can be a profile of nutrients or some attractant/repellent substances. To be specific, in this paper we considered a *Gaussian-type attractant/repellent profile* described by

$$\sigma(y) = -\frac{A_\sigma}{2} \exp\left(-\frac{\|y - c_\sigma\|^2}{l_\sigma}\right) + b_\sigma \tag{2}$$

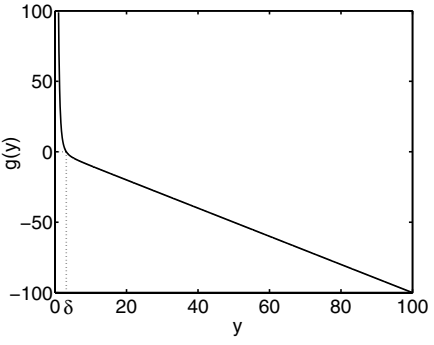


Fig. 1. Plot of function $g(\cdot)$. The attraction and repulsion balance at $y = \delta$, and attraction dominates for $y > \delta$ whereas repulsion dominates for $y < \delta$.

where $A_\sigma \in \mathbb{R},\ b_\sigma \in \mathbb{R},\ l_\sigma \in \mathbb{R}^+$, and $c_\sigma \in \mathbb{R}^n$, and $g(\cdot)$ is the attraction-repulsion function that describes a long-range attractive and short-range repulsive interaction between the individuals and takes the form

$$g(y) = -y\left(a - \frac{b}{\|y\|^4}\right), \tag{3}$$

where a, b are positive constants and $\|y\|^2 = y^\top y$. The first term $-ay$ represents the attraction, whereas the term $(by)/\|y\|^4$ represents the repulsion. For the case of $y \in \mathbb{R}^1$ and $a = 1,\ b = 100$, this function is shown in Fig. 1.

It can be easily seen that the function $g(\cdot)$ is attractive in a long distance and repulsive in a short distance. To be specific, $g(\cdot)$ changes its sign at the set of points $\Theta = \left\{ ||y|| = \sqrt[4]{\frac{b}{a}} = \delta \right\}$, where $g(\cdot) = 0$ and the attraction and repulsion hence balance, and the function is attractive (i.e. a dominates) for $\|x^i - x^j\| > \delta$ and repulsive (i.e. $b/||y||^4$ dominates) for $\|x^i - x^j\| < \delta$**.** For $||y|| = 0$ the function $g(y)$ has an infinite large value of repulsion. This prevents collisions in model (1) and thus $x^i(t) \neq x^j(t)$ for $i \neq j$ and for all $t \geq 0$. This is consistent with the fact that in reality any two individuals could not occupy the same position at the same time [10].

Remark 1. As noted in [9] that in (1) the term $-\nabla_{x^i}\sigma(x^i)$ represents the motion of swarm individuals toward areas with higher nutrient concentration and away from areas with higher toxic concentration. Note also that the model in (1) does not assume $w_{ij} = 1$ and therefore describes a general anisotropic system. Furthermore, it is assumed throughout the paper that the coupling matrix is irreducible, which implies that there are no isolated clusters in the swarm. It is clear from (3) that any two different swarm members could not occupy the same position at the same time. This is different from what considered in [7] and can guarantee collision avoidance in the swarms.

The aim of this paper is to present analytical and numerical results about the average motion, aggregation, stability and self-organized rotation of the swarm model (1) for Gaussian-type attractant/repellent profiles.

3 Quasi-reciprocal Swarms

We consider a class of nonreciprocal swarms whose coupling weights satisfying the so-called "detailed balance condition" specified as below.

Assumption 1 . *The swarm described in Eq. (1) satisfies the detailed balance condition in weights, that is, there are certain scalars* $\xi_i > 0$ $(i = 1, \cdots, M)$ *such that* $\xi_i w_{ij} = \xi_j w_{ji}$ *for all* i, j.

Since a swarm system often consists of a large number of individuals that may evolve in different manners, it is usually of interest to investigate collective behavior of the system rather than to ascertain detailed behavior of each individual—which may be very difficult or even impossible in general due to complex interactions among the large number of individuals. As a first step to this goal, it is simple and convenient to study the average motion of the swarm members. For a quasi-reciprocal swarm, the average motion of the swarm members can well be described by the "weighted center" of the swarm defined as below.

Definition 1. *The weighted center of the swarm members is* $\tilde{x} = \sum_{i=1}^{M} \xi_i x^i / \sum_{i=1}^{M} \xi_i$.

Since the function $g(\cdot)$ is symmetric, i.e., $g(-x) = -g(x)$ for all $x \in \mathbb{R}^n$, and $\xi_i w_{ij} g(x^i - x^j) = -\xi_j w_{ji} g(x^j - x^i)$. Then the average motion of the swarm members is described by

$$\frac{d\tilde{x}}{dt} = \left(\sum_{i=1}^{M} \xi_i\right)^{-1} \sum_{i=1}^{M} \xi_i \dot{x}^i = -\left(\sum_{i=1}^{M} \xi_i\right)^{-1} \sum_{i=1}^{M} \xi_i \nabla_{x^i} \sigma(x^i). \tag{4}$$

One can easily see that the profile has a unique global extremum (either a minimum or a maximum depending on the sign of A_σ) at $y = c_\sigma$. Its gradient at $y \in \mathbb{R}^n$ is

$$\nabla_y \sigma(y) = \frac{A_\sigma}{l_\sigma}(y - c_\sigma) \exp\left(-\frac{\|y - c_\sigma\|^2}{l_\sigma}\right). \tag{5}$$

Substituting (5) into (4), then $\frac{d\tilde{x}}{dt} = -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^{M} \xi_i\right)^{-1} \sum_{i=1}^{M} \xi_i (x^i - c_\sigma)\, exp\left(-\frac{\|x^i - c_\sigma\|^2}{l_\sigma}\right)$.

Now we define $e^i = x^i - \tilde{x}$ and $e_\sigma = \tilde{x} - c_\sigma$. Next we will analyze the motion of the weighted center $\tilde{x}$ and study the behavior of the swarm under the Gaussian-type profiles.

Theorem 1. *Consider the swarm modeled in Eqs. (1) and (3). Suppose that the* σ*-profile of the environment is described by Eq. (2) and Assumption 1 holds, we have as* $t \to \infty$ *that*

i) if $A_\sigma > 0$, *then* $\|e_\sigma(t)\| \leq \max_{i=1,\ldots,M} \|e^i(t)\| \triangleq e_{\max}(t)$;

ii) if $A_\sigma < 0$ *and* $\|e_\sigma(0)\| > e_{\max}(0)$, *(here we assume that* $x^i(0) \neq x^j(0)$ *for all pairs of individuals* (i,j), $j \neq i$, $1 \leq i,j \leq M$ *and therefore* $e_{\max}(0) > 0$*), then* $\|e_\sigma\| \to \infty$.

Proof. Let $V_\sigma = \frac{1}{2} e_\sigma^\top e_\sigma$. Using the fact that $x^i - c_\sigma = e^i + e_\sigma$, thus, its time derivative along the motion of the swarm is given by $\dot{V}_\sigma = -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)(x^i-c_\sigma)^\top e_\sigma = -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\|e_\sigma\|^2 - \frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right) e^{i\top} e_\sigma$. We will discuss the above equation as follows:

Case *a)* If $A_\sigma > 0$, then $\dot{V}_\sigma \leq -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\|e_\sigma\|^2 + \frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\|e^i\|\|e_\sigma\| \leq -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\left[\|e_\sigma\| - \frac{\sum_{i=1}^M \xi_i \exp\left(-\frac{x^i\ c_\sigma\ ^2}{l_\sigma}\right)\|e^i\|}{\sum_{i=1}^M \xi_i \exp\left(-\frac{x^i\ c_\sigma\ ^2}{l_\sigma}\right)}\right]\|e_\sigma\| \leq -\frac{A_\sigma}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)[\|e_\sigma\| - e_{\max}]\|e_\sigma\|$, where $e_{\max} = \max_{i=1,\ldots,M} \|e^i\|$. We can see that as long as $\|e_\sigma(t)\| > e_{\max}(t)$, the weighted center of the swarm will move to the minimum point c_σ. Thus, as $t \to \infty$, c_σ will be within the swarm.

Case *b)* If $A_\sigma < 0$, similar to the proof of the case *a)*, we have $\dot{V}_\sigma \geq \frac{|A_\sigma|}{l_\sigma}\left(\sum_{i=1}^M \xi_i\right)^{-1} \sum_{i=1}^M \xi_i \exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\|e_\sigma\|[\|e_\sigma\| - e_{\max}]$ which implies $\dot{V} > 0$, i.e., $\|e_\sigma\|$ will increase if $\|e_\sigma\| > e_{\max}$. Thus $\dot{V} > 0$ holds by hypothesis $\|e_\sigma(0)\| > e_{\max}(0)$. For any given large $D > 0$ and $\|e_\sigma(t)\| \leq D$ we can have $\exp\left(-\frac{\|x^i-c_\sigma\|^2}{l_\sigma}\right)\|e_\sigma\|[\|e_\sigma\| - e_{\max}] \geq \exp\left(-\frac{D^2+e_{\max}^2}{l_\sigma}\right) D[D - e_{\max}] > 0$ which implies that $\dot{V}_\sigma \geq \frac{|A_\sigma|}{l_\sigma} \exp\left(-\frac{D^2+e_{\max}^2}{l_\sigma}\right) D[D - e_{\max}] > 0$. Thus, by the Chetaev Theorem [11] we can conclude that $\|e_\sigma\|$ will exit the D-neighborhood of c_σ. This shows the desired result.

Next we will study the ultimate behavior of the individuals. Specifically, we define the set of equilibrium points of the swarm system as $\Omega_e = \{x : \dot{x} = 0\}$, where $x = \left\{x^{1\top}, \cdots, x^{M\top}\right\}^\top \in \mathbb{R}^{Mn}$. We will also prove the complete stability property of the swarm. We show that if $A_\sigma > 0$ in (2), then $x(t) \to \Omega_e$ as $t \to \infty$.

Definition 2. *The swarm modeled in Eqs. (1) and (3) is completely stable if every solution converges to an equilibrium point of the system.*

Theorem 2. *Consider the swarm modeled in Eqs. (1) and (3). Suppose that the* σ*-profile of the environment is a Gaussian-type profile described by Eq. (2) with* $A_\sigma > 0$ *and Assumption 1 holds, then the swarm is completely stable.*

Proof. With Assumption 1, we choose the Lyapunov function $V(x) = \sum_{i=1}^M \xi_i \sigma(x^i) + \frac{1}{2}\sum_{i=1}^{M-1}\sum_{j=i+1}^M \xi_i w_{ij}\left[a\|x^i - x^j\|^2 + \frac{b}{\|x^i-x^j\|^2}\right]$. Computing the gradient of $V(x)$

with respect to x^i yields $\nabla_{x^i} V(x) = \xi_i \nabla_{x^i} \sigma(x^i) - \sum_{j=1,j\neq i}^{M} \xi_i w_{ij} g(x^i - x^j) = -\xi_i \dot{x}^i$. Then we can take the time derivative of the Lyapunov function $V(x)$ along the motion of the system and obtain $\dot{V}(x) = [\nabla_x V(x)]^\top \dot{x} = \sum_{i=1}^{M} [\nabla_{x^i} V(x)]^\top \dot{x}^i = -\sum_{i=1}^{M} \xi_i \|\dot{x}^i\|^2 \leq 0$ for all t. By LaSalle's Invariance Principle [11] we can conclude that as $t \to \infty$ the state $x(t)$ converges to the largest positively invariant subset of the set defined by $\Omega = \{x : \dot{V}(x) = 0\} = \{x : \dot{x} = 0\} = \Omega_e$. This completes the proof.

Notice that the complete stability implies global convergence of the swarm system to its equilibrium point set. But the exact location and the size of this set cannot be known in general, especially for large number M of the swarm members, because the equations for equilibrium points are nonlinear. However, it is naturally expected that the swarm members will aggregate and form a cluster around the global minimum c_σ of the profile due to the long-range attraction effect. We will present such a result in the following. To do this, we need the following technical result.

Lemma 1. *Suppose that the swarm has only finite and isolated equilibrium points, then there exists a constant $\rho > 0$, such that $\|x^i - x^j\| \geq \rho$ holds for all i, j with $i \neq j$.*

Proof. Let the equilibrium points of the system in Eqs. (1) and (3) be $x_e^1, ..., x_e^M$. Take $\rho_0 = \min_{1\leq i,j\leq k} \{\rho_{ij} | \rho_{ij} = \|x_e^i - x_e^j\|, \forall i \neq j\}$. Then $\rho_0 > 0$ is a constant, and for all i, j with $i \neq j$, $\|x_e^i - x_e^j\| \geq \rho_0$. By Theorem 2, $x(t)$ converges to Ω_e. So, there exists a time $\overline{t} > 0$ such that, for all j , $\|x^j - x_e^j\| \leq \rho_0/4$ as $t > \overline{t}$. Therefore, we have $\|x^i - x^j\| \geq \|x_e^i - x_e^j\| - \|x^i - x_e^i\| - \|x^j - x_e^j\| \geq \rho_0/2 \triangleq \rho$. This shows the result.

Assumption 2 *There exists a constant $\bar{\sigma} > 0$ such that $\|\sqrt{\xi_i} \nabla_{x^i} \sigma(x^i)\| \leq \bar{\sigma}$ for all i.*

Theorem 3. *Consider the swarm modeled in Eqs. (1) and (3). Suppose that the σ-profile of the environment is a Gaussian-type profile described by Eq. (2) and Assumptions 1 and 2 hold, then as $t \to \infty$, all the members of the swarm will enter into a bounded region*

$$B_\mu = \left\{ x : \sum_{i=1}^{M} \xi_i \|x^i - \tilde{x}\|^2 \leq \mu^2 \right\}, \tag{6}$$

where $x = \left\{x^{1\top}, \cdots, x^{M\top}\right\}^\top \in \mathbb{R}^{Mn}$, $\mu = \frac{\xi_{\max}(b|||LW|||+2\theta\bar{\sigma}\rho^3)}{aM\gamma_2\rho^3}$ *with* $\xi_{\max} = \max_{1\leq i\leq M} \{\xi_i\} > 0$, $L = \text{diag}[\sqrt{\xi_1}, \cdots, \sqrt{\xi_M}]$, $|||LW||| = \sum_{i,j=1}^{M} \sqrt{\xi_i} w_{ij}$, $\theta \triangleq \left(\sum_{j=1}^{M} \xi_j\right)^{-1} \left(\sum_{j=1,j\neq i}^{M} \xi_j\right)$ *and γ_2 the second smallest real eigenvalue of the symmetric matrix $H = [h_{ij}]$ defined by*

$$h_{ij} = \begin{cases} -\xi_i w_{ij}, & i \neq j, \\ \sum_{l=1,l\neq i}^{M} \xi_i w_{il}, & i = j. \end{cases} \tag{7}$$

Moreover, for an arbitrary $\tilde{\varepsilon}$, all the swarm members out of the set $B_\mu(\tilde{x})$ will enter into its $\tilde{\varepsilon}$-neighborhood (i.e., $B_{\mu+\tilde{\varepsilon}}(\tilde{x})$) in a finite time bounded by $\tilde{T} = \frac{\xi_{\max}}{aM\gamma_2}\ln\left(\frac{\mu_0-\mu}{\tilde{\varepsilon}}\right)$, where $\mu_0 = \left(\sum_{i=1}^M \xi_i\|x^i(0)-\tilde{x}\|^2\right)^{\frac{1}{2}} > 0$.

Proof. Define $e^i = x^i - \tilde{x}$. From the definition of the weighted center $\tilde{x}$, we have

$$\begin{aligned}\dot{e}^i =& -\nabla_{x^i}\sigma(x^i) + \sum_{j=1,j\neq i}^M w_{ij}g(x^i-x^j) + \left(\sum_{j=1}^M \xi_j\right)^{-1}\sum_{j=1}^M \xi_j\nabla_{x^j}\sigma(x^j)\\ =& -a\sum_{j=1,j\neq i}^M w_{ij}(x^i-x^j) + b\sum_{j=1,j\neq i}^M w_{ij}\frac{x^i-x^j}{\|x^i-x^j\|^4} - \left[\nabla_{x^i}\sigma(x^i) - \left(\sum_{j=1}^M \xi_j\right)^{-1}\sum_{j=1}^M \xi_j\nabla_{x^j}\sigma(x^j)\right]\\ =& -a\sum_{j=1,j\neq i}^M w_{ij}(e^i-e^j) + b\sum_{j=1,j\neq i}^M w_{ij}\frac{e^i-e^j}{\|e^i-e^j\|^4} - \left[\nabla_{x^i}\sigma(x^i) - \left(\sum_{j=1}^M \xi_j\right)^{-1}\sum_{j=1}^M \xi_j\nabla_{x^j}\sigma(x^j)\right]. \quad (8)\end{aligned}$$

In order to estimate e^i, we choose the Lyapunov function $V(e) = \sum_{i=1}^M \xi_i V_i(e^i)$, where $V_i(e^i) = \frac{1}{2}e^{i\top}e^i$ and $e = [e^{1\top},\cdots,e^{M\top}]^\top$. Evaluating its time derivative along solution of the system (8) by Assumption 2 and making use of the fact that $\|x^i - x^j\| \geq \rho$ for all $i \neq j$ and that $\|e^i\| \leq \sqrt{2V(e)}$ $\forall i$, we can obtain $\dot{V}(e) = \sum_{i=1}^M \xi_i e^{i\top}\left[-a\sum_{j=1,j\neq i}^M w_{ij}(e^i-e^j) + b\sum_{j=1,j\neq i}^M w_{ij}\frac{e^i-e^j}{\|e^i-e^j\|^4}\right] - \sum_{i=1}^M \xi_i\left[\nabla_{x^i}\sigma(x^i) - \left(\sum_{j=1}^M \xi_j\right)^{-1}\sum_{j=1}^M \xi_j\nabla_{x^j}\sigma(x^j)\right]^\top e^i \leq -a\sum_{i=1}^M\sum_{j=1,j\neq i}^M w_{ij}\xi_i e^{i\top}(e^i - e^j) + b\sum_{i=1}^M\sum_{j=1,j\neq i}^M \xi_i w_{ij}\frac{\|e^i-e^j\|\|e^i\|}{\|e^i-e^j\|^4} + 2\bar{\sigma}\left(\sum_{j=1}^M \xi_j\right)^{-1}\left(\sum_{j=1,j\neq i}^M \xi_j\right)\sum_{i=1}^M\|\sqrt{\xi_i}e^i\| \leq -a\sum_{i=1}^M\left(\sum_{j=1,j\neq i}^M \xi_i w_{ij}\right)e^{i\top}e^i + a\sum_{i=1}^M\left(\sum_{j=1,j\neq i}^M \xi_i w_{ij}\right)e^{i\top}e^j + \sqrt{2}b|||LW|||\rho^{-3}V^{\frac{1}{2}}(e) + 2\sqrt{2}\bar{\sigma}\left(\sum_{j=1}^M \xi_j\right)^{-1}\left(\sum_{j=1,j\neq i}^M \xi_j\right)V^{\frac{1}{2}}(e) = -ae^\top(H\otimes I)e + \sqrt{2}b|||LW|||\rho^{-3}V^{\frac{1}{2}}(e) + 2\sqrt{2}\theta\bar{\sigma}V^{\frac{1}{2}}(e)$, where $\theta \triangleq \left(\sum_{j=1}^M \xi_j\right)^{-1}\left(\sum_{j=1,j\neq i}^M \xi_j\right)$, $H\otimes I$ is the Kronecker product of H as defined in (7) and I the identity matrix of order n. To get further estimate of $\dot{V}(e)$, we only need to estimate the term $e^\top(H\otimes I)e$.

We can easily conclude that $\gamma = 0$ is an eigenvalue of H and $u = (l,\cdots,l)^\top$ with $l \neq 0$ is the associated eigenvector. Moreover, because H is symmetric with all $h_{ij} \leq 0$ for all $i \neq j$ and W is irreducible (so is H), it follows from matrix theory [8] that $\gamma = 0$ is a simple eigenvalue and all the rest eigenvalues of $-H$ are real and negative . Therefore, we can order the eigenvalues of H as $0 = \gamma_1 < \gamma_2 \leq \cdots \leq \gamma_n$. Also it is known that the identity matrix I has an n multiple eigenvalues $\nu = 1$ and n independent eigenvectors

$$d^1 = \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix},\quad d^2 = \begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix},\quad \cdots,\quad d^n = \begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}.$$

By matrix theory [8], the eigenvalues of $H \otimes I$ are $\gamma_i \nu = \gamma_i$ (n multiple for each i) and the corresponding eigenvectors are $u^i \otimes d^j$. It is clear that, because $H \otimes I$ is symmetric, the n^2 eigenvectors $u^i \otimes d^j$ are linearly independent. So, if $e^\top (H \otimes I)e = 0$ then e must lie in the eigenspace of $H \otimes I$ spanned by eigenvectors $u \otimes d^i$ corresponding to the zero engenvalue, that is, $e^1 = e^2 = \cdots = e^M$. This occurs only when $e^1 = e^2 = \cdots = e^M = 0$. But this is impossible for the swarm model under consideration, because it implies that the M individuals occupy the same position at the same time. Hence, for any solution x of system (1), e must be in the subspace spanned by eigenvectors of $H \otimes I$ corresponding to the nonzero eigenvalues. Then, $e^\top (H \otimes I)e \geq \gamma_2 \|e\|^2 = 2\gamma_2 V(e)$. From above, we have $\dot{V}(e) \leq -\frac{2aM\gamma_2}{\xi_{\max}} V(e) + \sqrt{2}b|||LW|||\rho^{-3} V^{\frac{1}{2}}(e) + 2\sqrt{2}\theta\bar{\sigma} V^{\frac{1}{2}}(e) = -\frac{2aM\gamma_2}{\xi_{\max}} V(e) + \sqrt{2}\left\{b|||LW|||\rho^{-3} + 2\theta\bar{\sigma}\right\} V^{\frac{1}{2}}(e) < 0$ whenever $V(e) > \left(\frac{\xi_{\max}(b|||LW|||+2\theta\bar{\sigma}\rho^3)}{\sqrt{2}aM\gamma_2\rho^3}\right)^2$. This shows that e^i converges to the region $B_\nu(\bar{x})$ as $t \to \infty$, where $\mu = \xi_{\max}(b|||LW||| + 2\theta\bar{\sigma}\rho^3)/aM\gamma_2\rho^3$. Since i is arbitrary, the result holds for all the members. From above, we also have $\dot{V} \leq -\frac{2aM\gamma_2}{\xi_{\max}} V(e) + \sqrt{2}\left(b|||LW|||\rho^{-3} + 2\theta\bar{\sigma}\right) V^{\frac{1}{2}}(e)$ for $\sum_{i=1}^M \xi_i \|e^i\|^2 \geq \mu^2$. Then $dV^{\frac{1}{2}} \leq -\frac{aM\gamma_2}{\xi_{\max}}\left(V^{\frac{1}{2}} - \frac{\xi_{\max}(b|||LW|||+2\theta\bar{\sigma}\rho^3)}{\sqrt{2}a\lambda_2\rho^3}\right) dt = -\frac{aM\gamma_2}{\xi_{\max}}\left(V^{\frac{1}{2}} - \frac{\mu}{\sqrt{2}}\right) dt$, so $\ln\left(\frac{\sqrt{2V(t)}-\mu}{\sqrt{2V(0)}-\mu}\right) \leq -\frac{aM\gamma_2}{\xi_{\max}} t$ for $\sum_{i=1}^M \xi_i \|e^i\|^2 \geq \mu^2$. Suppose a member i enters into the $\tilde{\varepsilon}$-neighborhood of the region $B_\nu(\tilde{x})$ at time $\tilde{T}$, then $V(\tilde{T}) = \frac{1}{2}(\tilde{\varepsilon} + \mu)^2$, and from above, $\ln\left(\frac{\tilde{\varepsilon}}{\mu_0 - \mu}\right) \leq -\frac{aM\gamma_2}{\xi_{\max}}\tilde{T}$, i.e. $\tilde{T} \leq \frac{\xi_{\max}}{aM\gamma_2} \ln\left(\frac{\mu_0-\mu}{\tilde{\varepsilon}}\right)$. Therefore, any solution of system (8) will eventually enter into and remain in $B_{\mu+\tilde{\varepsilon}}(\bar{x})$.

Theorem 4. *Consider the swarm modeled in Eqs. (1) and (3). Suppose that the σ-profile of the environment is a Gaussian-type profile described by Eq. (2) and that Assumptions 1 and 2 hold, then as $t \to \infty$, the following hold:*

i) if $A_\sigma > 0$, then all individuals will enter $B_{2\mu}(c_\sigma)$;

ii) if $A_\sigma < 0$ and $\|e_\sigma(0)\| \geq e_{\max}(0)$, then for any fixed $D > 0$ all individuals will exits $B_D(c_\sigma)$.

For $A_\sigma > 0$, from Theorem 3 we know that the swarm will have a maximum size of μ, that is, $\|e_\sigma\| \leq \mu$ for all i, and we can also see that the swarm center will converge to the $e_{\max}$ and therefore to the μ-neighborhood of c_σ, that is, $\|e_\sigma\| \leq e_{\max} \leq \mu$ from Theorem 1 we can obtain the 2μ when $A_\sigma > 0$ combining the two bounds.

Remark 2. The above discussions explicitly show the effect of the coupling matrix W on aggregation and cohesion of the swarm. Figs. 2,3 give numerical illustrations of the results with parameters $M = 10$, $a = 1$, $b = 20$, and the extremum of the Gaussian-type profile chosen at $c_\sigma = [-2, 12]^\top$ and magnitude $A_\sigma = \pm 0.01$. Figs. 2,3 show the paths of the swarm individuals and the swarm weighted center with the Gaussian-type profile when $A_\sigma > 0$ and $A_\sigma < 0$, respectively. From these figures we can easily see that the simulations are consistent with the theoretical results, i.e., the swarm weighted center $\tilde{x}$ converges to the

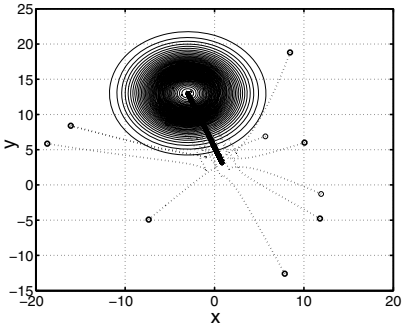


Fig. 2. The paths of the weighted center and the individuals when $A_\sigma > 0$

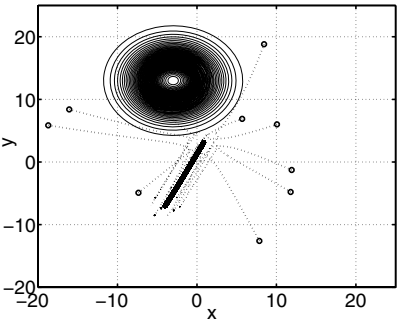


Fig. 3. The paths of the weighted center and the individuals when $A_\sigma < 0$

minimum of the Gaussian-type profile c_σ when $A_\sigma > 0$ and diverges from the maximum when $A_\sigma < 0$.

Remark 3. Note that when $\xi_i = \xi_j$ for all i, j, we have $w_{ij} = w_{ji}$, which represents a reciprocal swarm. It is a specific case of the model we considered above.

4 Oscillatory Motion in General Nonreciprocal Swarms

Now we give some simulations of oscillatory motion in nonreciprocal swarms. It is usually difficult to analytically investigate collective behavior of a general nonreciprocal swarm. To obtain some ideas about the motion of general nonreciprocal swarms, we have performed some numerical simulations for the swarm model in (1) and (3). Particularly, we observed that in certain cases, the swarm may exhibit more complex oscillatory behavior rather than convergence.

Figs. 4,5 show the results of our simulations with parameters $M = 10$, $a = 1, b = 20$, and the extremum of the Gaussian-type profile chosen at $c_\sigma =$

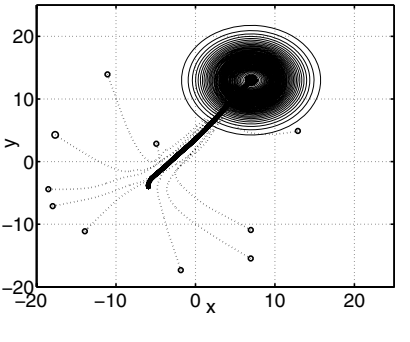


Fig. 4. The paths of the weighted center and the individuals when $A_\sigma > 0$

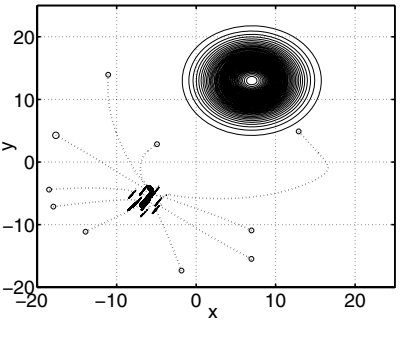


Fig. 5. The paths of the weighted center and the individuals when $A_\sigma < 0$

$[7, 12]^\top$. The coupling matrix W is generated randomly and scaled such that $w_{ii} = 0,\ 0 < w_{ij} < 1$ for all $i \neq j$. For its large size, we do not include the explicit value of W here. It is available upon request. Figs. 4,5 show the paths of the swarm individuals and the swarm weighted center with the Gaussian-type profile when $A_\sigma > 0$ and $A_\sigma < 0$, respectively. From Figs. 4,5, we can observe that at the beginning phase of the simulation, the swarm members gradually aggregate and form a cohesive cluster. Then they move in the same direction as a group while keeping mutual spacing, and eventually evolve into a rotatory motion as time processes.

5 Conclusions

We have studied an anisotropic swarm model and shown that it can capture the basic feature of aggregation, cohesion and complete stability of the swarm under detailed balance condition and Gaussian-type attractant/repellent profiles. We also gave estimates on the size of the cohesion cluster. For general nonreciprocal swarms, our numerical results shows that more complex self-organized oscillation may occur in the systems. Our model can guarantee collision avoidance in the swarm. The results provide further insight into the effect of the interaction pattern on collective motion in a swarm.

References

1. C. M. Breder, "Equation descriptive of fish schools and other animal aggregations," Ecology **35** (1954) 361–370
2. R. Arkin, Behavior–Based Robotics, Cambridge MA: MIT Press, 1998.
3. J. R. T. Lawton, R. W. Beard, and B. J. Young, "A decentralized approach to formation maneuvers," IEEE Trans. Robot Automat. **19** (2003) 933–941
4. A. Okubo, "Dynamical aspects of animal grouping: swarms, schools, flocks, and herds," Adv. Biophys. **22** (1986) 1–94
5. A. Czirok, H. E. Stanley, and T. Vicsek, "Spontaneously ordered motion of self-propelled particles," J. Phys. A: Math., Gen. **30** (1997) 1375–1385
6. J. Toner and Y. Tu, "Flocks, herds, and schools: A quantitative theory of flocking," Phys. Rev. E **58** (1998) 4828–4858
7. V. Gazi and K. M. Passino, "Stability analysis of swarm," IEEE Trans. Automat. Contr. **48** (2003) 692–697
8. T. Chu, L. Wang, and S. Mu, "Collective behavior analysis of an anisotropic swarm model," *Proc. 16th Int. Symp. Math. Theor. Networks Syst.*(MTNS 2004), Paper ID: REG–345, Leuven, Belgium, July 2004, pp. 1–14.
9. V. Gazi and K. M. Passino, "Stability analysis of social foraging swarms, IEEE Trans. Syst., Man, and Cybernetics, Part B: Cybernetics **34** (2004) 539–557
10. B. Liu, T. Chu, L. Wang, and Z. Wang, "Swarm Dynamics of A Group of Mobile Autonomous Agents," Chin. Phys. Lett. **22** (2005) 254–257
11. H. Khalil, *Nonlinear Systems*, 2nd ed., Prentice–Hall, Inc., Upper Saddle River, 1996

Evolving Annular Sorting in Ant-Like Agents

André Heie Vik

Complex Adaptive Organically-Inspired System Group (CAOS),
The Norwegian University of Science and Technology,
Trondheim, Norway
andrehei@idi.ntnu.no

Abstract. This paper describes an evolutionary approach to the design of controllers for a team of collective agents. The agents are able to perform ant-like annular sorting, similar to the sorting behaviour seen in the ant species Temnothorax albipennis. While most previous research on ant-like sorting has used hard-wired rules, this study uses neural network controllers designed by artificial evolution. The agents have very simple and purely local sensory capabilities, and can only communicate through stigmergy. Experiments are performed in simulation. The evolved behaviours are presented, analyzed, and compared to previous research on ant-like annular sorting. The results show that artificial evolution is able to create efficient, simple, and scalable controllers able to perform annular sorting of three object types.

1 Introduction

Social insects are known for their ability to perform complex tasks without need for direct communication. This is achieved through self-organisation, in which order at the global level of a system emerges from the interactions between the system's lower level components. One form of self-organisation is stigmergy, a concept originally used to explain building behaviour in termites. In stigmergy, agents communicate indirectly through the work performed on the environment, involving both positive and negative feedback [1]. Recently, there has been significant scientific interest in trying to transfer these principles to engineering domains such as robotics and algorithms. The motivation for this interest is the simplicity, robustness, and scalability of the principles employed by social insects [2]. At the same time, biologists are increasingly interested in robotics and artificial life simulations as a means for testing out hypotheses on animal behaviour and self-organisation [3].

The behaviour-based robotics paradigm [4] requires the designer to break down the complex system behaviour into a set of simple basic behaviours. This means that the designer must make assumptions about the kind of behaviour needed to perform the given task. Collective systems are made up of many interdependent parts, making it very difficult to do this decomposition [5]. On the other hand, in evolutionary robotics [6] the structure of the control system emerges from the interactions between the robot and the environment through

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 594–603, 2005.

a process of natural selection. This makes evolutionary robotic methods well suited for designing controllers for collective systems.

The work described in this paper is inspired by brood sorting in the ant species Temnothorax albipennis. Artificial evolution is used to design neural network controllers for a homogenous team of simple, simulated agents which are able to sort three object types in annular bands, similar to the way T. albipennis ants sort their brood. T. albipennis ants sort their brood in concentric rings, based on the developmental stage of the brood. This sorting behaviour has been termed annular sorting, and it is a much studied, though still not fully explained example of self-organisation through stigmergy [7]. Deneubourg et. al. [8] provided the first example of ant-like robots that were able to sort objects of different types in simulation. Beckers et. al. [9] implemented clustering of one object type with minimalist robots controlled by a simple rule set. Melhuish et. al. later extended this work to include sorting of multiple object types [10,11]. Recently, Wilson et. al. [12] achieved ant-like annular sorting with real robots.

A property shared by all the work described above is that each relies on hard-wired rules, although Wilson et. al. used a combined leaky integrator tuned by genetic algorithms to set some parameters of their rule set. In parallel with the work described in this article, Hartmann [13] evolved neural network controllers for a swarm of agents. The main difference between this work and Hartmann's work is that while Hartmann's agents have very detailed sensory information and reside in a grid world, the agents described in this article have very limited sensory capabilities and are simulated in continuous space and time. In this respect, this work has more in common with robotic experiments, although it does not claim to be a realistic simulation of real robots.

2 Experimental Framework

2.1 Environment and Task

The simulated environment consists of a flat plane with walls placed in a octagonal shape, making an arena similar to that used by Melhuish et. al. [10]. Each wall is 42 units long, giving a ratio between agent size and arena size of 1800, similar to the ratio between ant size and nest size in T. albipennis colonies [10]. At the start of a trial, 42 brood objects, 14 of each of 3 types, are placed uniformly across the plane, as illustrated in fig. 3(a). Six agents are placed in random starting positions, facing in random directions. One neural network controller is cloned to all agents, making the team homogenous. The simulation is terminated after 400 simulated seconds. The agents are supposed to sort the different types of brood objects in annular bands.

Experiments were performed in simulation, using the BREVE simulation environment [14]. BREVE simulates continous space and time, handles collision detection and provides rich 3D visualisation. As BREVE simulates continuous time, there are no discrete time steps in the simulation. However, the controllers of the agents are activated each 0.05 seconds (an iteration). This value can be interpreted as the reaction time of the agents to changes in their environment.

Between activations, multiple collisions and movement of world objects can occur, but the agents can only sense and act at each activation. Brood hits are accumulated between iterations.

2.2 Ant Agents and Brood Objects

The agents are circular, with a radius of 1.5 units. Sensory capabilities are minimal, similar to that of other work on ant-like sorting. This has both a biological and an engineering motivation; the limited sensory capabilities of real ants, and the wish to solve the task with the simplest agents possible. In order to simulate the uncertainty of real robot sensors, noise is added to sensor inputs and actuator outputs with a probability of 0.01.

Actuators. The agents move with constant velocity, and are able to turn in any given direction. At the front of the agent there is a gripper, which can be turned on and off. When the agents hit an object within a 45° angle of the movement direction the gripper is activated, the object is gripped. Gripped objects are moved in front of the robot as long as the gripper is on.

Sensors. When an agent collides with an object, it can sense the type of object hit; wall, other agent, type of brood. The type of brood object held in the gripper is also sensed.

The brood objects are the sorting materials used in the simulation. Like the ant agents, they are circular, with a radius of 1.5 units. Each brood object belongs to a type, which ant agents can recognize.

2.3 Controller

Each agent is controlled by an artificial neural network. At each simulator iteration, the network is activated with inputs accumulated since the last iteration. Movement direction and gripper status are then updated using the new network output.

The artificial neural network used (fig. 1) is a simple recurrent network [15] with 5 fully recurrent connected hidden nodes. There are 9 input nodes, 2 output

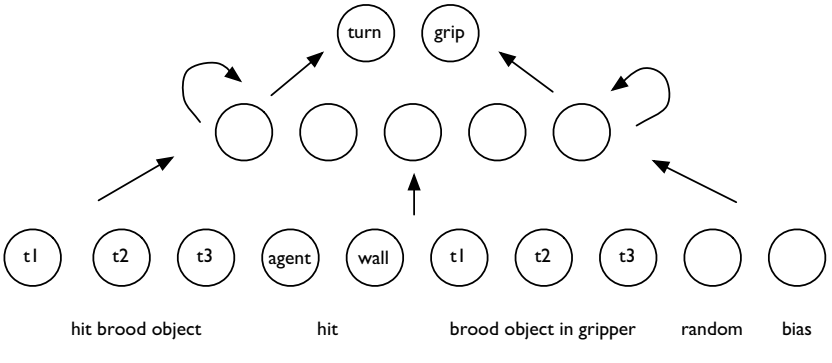


Fig. 1. Neural Network Architecture. Fully connected simple recurrent network.

nodes, and 1 bias node. In addition to sensor inputs from the agent described above, a random variable with a normal distribution $N(0,1)$ is input to the network. This makes it possible to evolve nondeterministic behaviours. The network uses the logistic activation function for all hidden nodes and the turn angle output node. The gripper status output node uses a step activation function.

2.4 Fitness

The fitness function is based on the annular sorting performance measure proposed by Wilson et. al. [12] They argue that the compactness and separation metrics are the two most useful components of the performance measure in their study of sorting mechanisms. Based on this, these two metrics were selected as fitness function for this experiment.

Compactness is the average radial distance from the centre of the structure (the average vector of all brood objects), normalized to (0,100), such that 100 represents a perfectly packed structure, and 0 represents poor compactness. $optD(t)$ represents the optimum mean distance from the centre for the total number of objects, t, when they are optimally packed inside a circle [16]. l is the side length of the arena. $\bar{x}$ represents the mean distance to the centre of the structure. Equation 1 shows the calculation of the compactness metric.

$$C = 100 * \left(1 - \frac{((\bar{x} - optD(t))}{(1.5l - optD(t)))}\right) \tag{1}$$

Separation is the percentage of brood objects that infringe on the 'home zone' of another object type. To calculate the separation metric, objects are sorted according to type, then the radial distance from the centre of the structure is calculated. The distance between the upper and lower quartiles of each type is the 'home zone' of the object type. The metric is then calculated using four different counts. For the central type objects, the number of objects C^1_{glq}, which have a radial distance to the centre greater than the lower quartile of any other type is counted. For the outer-most type objects, the number of objects O^m_{luq}, which have a radial distance to the centre less than the upper quartile of any other type is counted. For each of the intermediate object types, two counts are performed. I^c_{glq} is the number of objects which have a radial distance to the centre greater than the lower quartile of any object type further from the centre. I^c_{luq} is the number of objects which have a radial distance to the centre less than the upper quartile of any object type closer to the centre. n_c is the number of objects per type, m is the number of object types. Equation 2 shows the calculation of the metric from the counts described above.

$$S = 100 * \left(1 - \frac{C^1_{glq} + O^m_{luq} + \sum_{c=2}^{c=m-1} \frac{(I^c_{glq} + I^c_{luq})}{2}}{\sum_{c=1}^{c=m} n_c}\right) \tag{2}$$

The fitness of a controller is the weighted sum of the two components, measured at the end of each run. The constant w determines the relative weights

of the two components. Although the environment is identical in each run, the random nature of the sorting process means that fitness varies between runs. Due to the computational costs of the simulations, each controller is tested for only two simulation runs. The average fitness of these two runs is recorded as fitness for the controller.

$$F = wC + (1 - w)S \tag{3}$$

2.5 Evolutionary Algorithm

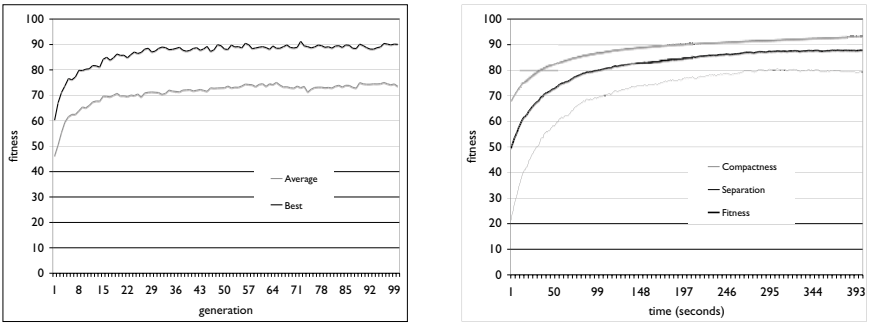
The controllers are evolved from an initial random population of 100 individuals. The best 10 genotypes of each generation are selected as parents for the next generation. One unchanged copy and 9 mutated copies of each parent genotype is copied to the new generation. Recombination is not used. The genotype is represented as a vector of real values, each encoding one connection weight (input, bias, recurrent, and output connections). Mutation is performed by adding a random variable with a normal distribution $N(0, s)$ to each connection weight. In the experiments described here, $s = 0.2$. The initial population is initialized with random connection weights with a normal distribution $N(0, 1)$.

3 Results

The evolutionary experiment described above was replicated 10 times, each time for 100 generations. Each replication started from a different randomized population. 100 generations took about 4 days to complete on one processor of a Apple Power Mac G5 2 GHz workstation. The fitness development is shown in fig. 2(a). Good fitness is achieved after about 30 generations. Later improvements were small in terms of fitness, but in some cases gave large improvements in the observed (qualitative) level of annularity.

The weighting between the two fitness metrics was crucial to the level of annularity achieved. With equal weight on both metrics (w=0.5), the controllers achieving the best fitness were those that did not move the outer type at all. This is caused by the high risk of affecting separation when compactness is improved. By increasing the weight of the compactness component slightly (w=0.6), better performance was achieved. With higher weight on the compactness component, it became more advantageous to move and cluster the outer type as well.

The best evolved controller was verified for 100 simulator runs. Average fitness was 88.8, with a standard deviation of 3.0. The separation component had a standard deviation of 6.8, compared to 1.3 for the compactness component. A plot of the two fitness metrics and total weighted fitness over time (fig. 2(b)) shows that good compactness and separation is achieved after about 150 seconds, with very small subsequent changes. Though there are differences in performance, this general pattern is seen in the best evolved controllers of all replications. In order to investigate the generality of the evolved controllers, the best individual was tested 100 times with different random brood layouts. The average compactness and separation scores were about one standard deviation below the results for the uniform layout.



(a) Performance over 100 generations, averaged over all 10 replications. Black line represents best fitness, gray line represents average fitness.

(b) Team performance with 6 agents (fitness over time) for the best individual, averaged over 100 runs. Black line represents total fitness, dark gray line compactness, and thin black line separation.

Fig. 2.

3.1 Analysis of the Evolved Behaviours

Figure 3 shows the development of the sorted structure during a simulation run with the best evolved controller. In the first 100 seconds of the run, objects of the innermost type are clustered in one large, central cluster, while the intermediate type objects are clustered in several smaller clusters further from the centre. Objects of the outermost type are mostly left alone. In the next 100 seconds, the objects of the intermediate type are placed around the central cluster. Finally, objects of the outermost type are scattered in a broad belt around the central cluster, in which separation is constantly improved. Fig. 4(a) shows how well the different types are separated at the end of the run. The different types are relatively well separated, with the 'home zones' (interquartile ranges) perfectly separated.

Observations of the best evolved controllers show that they exploit a few simple mechanisms to accomplish the sorting described above. Similar behaviour emerged in 9 out of 10 replications, with the overall best results in the first replication, which this analysis is based on. The agents move in circles, the size of which depends on whether or not there is an object in the gripper. This enables individual positioning of the different object types. With no object in the gripper, the agents move in large circles limited by the outer walls, sometimes making a turn into the centre of the arena. This can be described as a global search behaviour, in which the agents move around the arena searching for unsorted objects. With an object in the gripper, the movement changes to smaller circles, with the smallest circles observed for the inner object type. This local search behaviour, combined with a higher probability of dropping objects next to other objects of the same type, makes it possible to sort and cluster objects.

Additionally, the different object types are treated differently in terms of movement distance and time of pickup. Analysis of brood carrying (table 1) show that objects of the inner type are moved first, and carried for the longest average distance. The intermediate type objects are carried shorter and later, but also more than twice as frequently as the inner type. The outermost objects are moved for very short distances, but with an even higher frequency. In fact, the outermost object type is always released from the gripper at the next iteration. The difference in time of movement for the different types is mostly caused by the fact that there is a lower probability of hitting objects in the centre than in the periphery when moving in large circles (global search). This makes objects clustered in the centre less likely to be picked up and moved. When testing the evolved controllers without the inner object type, the intermediate object type was clustered in a central cluster. This occured later than central object type clustering would have happened, indicating that compact annular sorting of the intermediate type is dependent on the early formation of a central cluster of the central object type.

An interesting property of the evolved controllers is that the structure very often is built in the centre of the arena, even though this does not give any additional fitness. A reason for this may be that a central structure is less likely to be hit and damaged, given the movement patterns described above. Another possible explanation may be that agents use the walls in combination with the described movement patterns as a form of simple navigation.

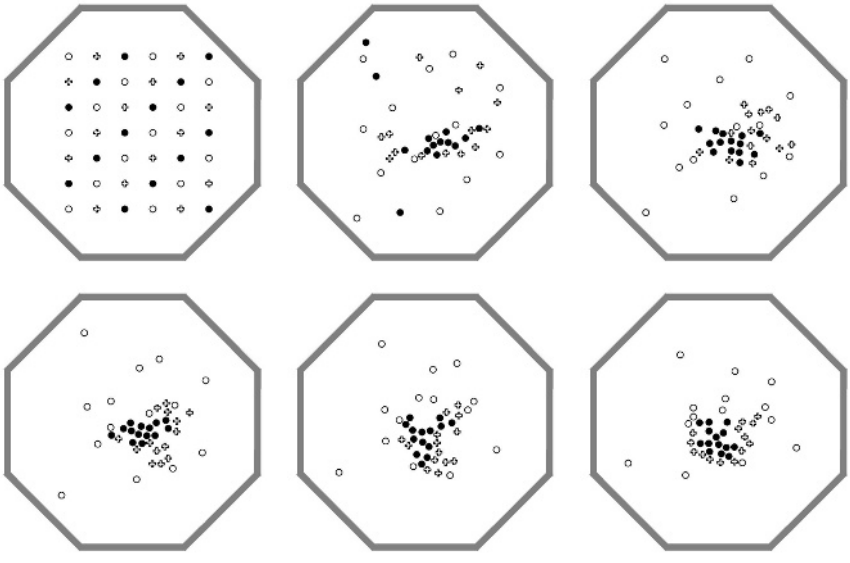
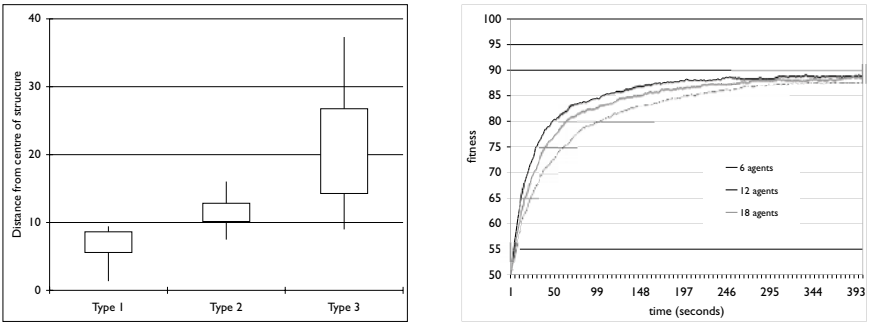


Fig. 3. Example of simulation progress, $t = 0, 50, 100, 200, 300, 400$. Legend; inner: black circle; intermediate: cross; outer: white circle. Fitness at $t = 400$: 90.8.

Table 1. Brood carrying statistics for 6 agents, grouped by object type. Average distance is average of moved distance. Average time is average of simulation time when the object was released. N is number of moved objects. Averaged over 100 runs.

Object type	Avg distance	Med distance	Avg time	Med time	N
1	9.1	6.7	137.2	97.0	351
2	2.0	0.39	166.7	145.3	736
3	0.3	0.26	203.9	202.6	1820
all types	1.7	0.30	187.3	180.3	2857



(a) Separation: Box and whisker plot of radial displacement for the different types at the end of the run shown in fig. 3. Line represents maximum and minimum distance, box represents interquartile range.

(b) Scalability: team performance (fitness over time) for the best evolved controller, averaged over 100 runs. Thin black line represents 6 agents, black line represents 12 agents, dark gray line 18 agents.

Fig. 4.

3.2 Scalability

A key issue in collective systems is scalability. To investigate the scalability of the evolved controllers, the best evolved neural network controller was tested with group sizes of 6, 12 and 18 agents for 100 simulation runs each. Each run lasted for 400 seconds. Fitness was recorded every second. The performance gain with 12 agents was nearly linear compared to the 6 agent case (fig. 4(b)). 18 agents gave poorer performance than 12 agents. The reason for this is probably that the ratio between agents and brood is too high with 18 agents and 42 brood objects. The density of agents in the arena is also very high with 18 agents, making interference a larger problem. With a larger arena and more brood objects, it would very likely be possible to scale to larger teams.

4 Discussion

This study describes an experiment on designing controllers for ant-like agents performing collective annular sorting. The mechanisms behind annular sorting

in ants are still not fully explained, and there have been few successful attempts to reproduce it in simulation. Most previous work on ant-like sorting has used hard-wired rules. This study takes another approach, by using neural network controllers designed by artificial evolution. Annular sorting of three object types was accomplished in a complex and noisy environment by agents with very limited sensory capabilities. The results show that artificial evolution is able to create efficient, simple, and scalable controllers for this challenging collective task.

Although this study concerns agents, not physical robots, it shares many properties with swarm robotics. Dorigo and Sahin [17] have identified four criteria for distinguishing swarm robotics from other multi-robot studies:

1. The study should be relevant for the coordination of large numbers of robots
2. The system being studied should consist of relatively few homogenous groups of robots, and the number of robots in each group should be large
3. The robots used in the study should be simple and incapable with respect to the task considered, so that cooperation is needed to complete the task
4. The robots used in the study should have only local and limited sensing and communication abilities

This study fully meets criteria 2 and 4. For criteria 1, good scalability has been shown for team sizes up to 12 agents. With larger teams, it is likely that the agent/brood ratio and size of arena is limiting the performance. Further experiments on the relationship between arena size, number of objects, and number of agents are needed to better answer these questions. Criteria 3 is not fully met. A single agent could complete the task alone, but performance does improve with cooperation.

In future work, it would be useful to verify the method used here in other, more complex settings. Full physical simulation could yield simpler behaviours, as well as making it more likely that the controllers could be used in real robots. More object types would make it possible to fully replicate the pattern created by ants. By changing the size of the arena, it could be revealed if the evolved behaviours depend on the environment in order to successfully sort the objects.

References

1. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-Organization in Biological Systems. Princeton University Press (2001)
2. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence: from natural to artificial systems. Oxford University Press (1999)
3. Webb, B.: What does robotics offer animal behaviour? Animal Behaviour **60** (2000) 545–558
4. Arkin, R.C.: Behavior-Based Robotics. MIT Press (1998)
5. Baldassarre, G., Nolfi, S., Parisi, D.: Evolving mobile robots able to display collective behaviour. Artificial Life **9** (2003) 255–267

6. Nolfi, S., Floreano, D.: Evolutionary Robotics – The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press (2000)
7. Sendova-Franks, A.B., Scholes, S.R., Franks, N.R., Melhuish, C.: Brood sorting by ants: two phases and differential diffusion. Animal Behaviour **68** (2004) 1095–1106
8. Deneubourg, J., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chretien, L.: The dynamics of collective sorting: Robot-like ants and ant-like robots. In: From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior, MIT Press (1991) 353–363
9. Beckers, R., Holland, O.E., J.L, D.: From local actions to global tasks: Stigmergy in collective robotics. In: Artificial life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, MIT Press (1994) 181–189
10. Melhuish, C., Holland, O.E., Hoddell, S.: Collective sorting and segregation in robots with minimal sensing. In: From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior, MIT Press (1998) 465–470
11. Melhuish, C., Wilson, M., Sendova-Franks, A.: Patch sorting: Multi-object clustering using minimalist robots. In: Advances in Artificial Life: Proceedings of 6th European Conference (ECAL 2001), Springer (2001) 543–552
12. Wilson, M., Melhuish, C., Sendova-Franks, A.B., Scholes, S.: Algorithms for building annular structures with minimalist robots inspired by brood sorting in ant colonies. Autonomous Robots **17** (2004) 115–136
13. Hartmann, V.: Evolving agent swarms for clustering and sorting. In: Proceedings of the Genetic and Evolutionary Computation Conference: GECCO 2005, ACM Press (2005) In press.
14. Klein, J.: breve: a 3d simulation environment for the simulation of decentralized systems and artificial life. In: Artificial Life VIII: Proceedings of the 8th International Conference on Artificial Life, MIT Press (2002)
15. Elman, J.: Finding structure in time. Cognitive Science **14** (1990) 179–211
16. Graham, R.L., Lubachevsky, B.D., Nurmela, K., Östergård, P.: Dense packings of congruent circles in a circle. Discrete Mathematics **181** (1996) 139–154
17. Dorigo, M., Şahin, E.: Swarm robotics – special issue editorial. Autonomous Robots **17** (2004) 111–113

Flocking Control of Multiple Interactive Dynamical Agents with Switching Topology via Local Feedback

Hong Shi[1], Long Wang[1], Tianguang Chu[1], and Minjie Xu[2]

[1] Intelligent Control Laboratory, Center for Systems and Control,
Department of Mechanics and Engineering Science,
Peking University, Beijing 100871, P. R. China
{hongshi, longwang, chutg}@pku.edu.cn
[2] Department of Information Engineering,
Shandong Water Polytechnic, Shandong 276826, P. R. China
sdsxxmj@163.com

Abstract. This paper considers a group of mobile autonomous agents moving in the space with point mass dynamics. We investigate the dynamic properties of the group for the case that the topology of the neighboring relations between agents varies with time. We introduce a set of switching control laws and show that the desired stable flocking motion can be achieved by using them. The control laws are a combination of attractive/repulsive and alignment forces, and the control law acting on each agent relies on the state information of its neighbors and the external reference signal. By using the control laws, all agent velocities asymptotically approach the desired velocity, collisions are avoided between the agents, and the final tight formation minimizes all agent potentials. Finally, numerical simulations are worked out to further illustrate our theoretical results.

1 Introduction

Stimulated by the simulation results in [1], Tanner *et al.* [2] considered a group of mobile agents moving in the plane with double integrator dynamics. They introduced a set of control laws that enable the group to generate stable flocking motion, and provided theoretical justification. From [3], it is easy to see that these control laws cannot regulate the final speed and heading of the group. On the other hand, in reality, the motion of the group sometimes is inevitably influenced by some external factors. In some cases, the regulation of agents has certain purposes such as achieving desired common speed and heading, or arriving at a desired destination. Therefore, the cooperation/coordination of multiple mobile agents with some virtual leaders is an interesting and important topic. There have been some papers dealing with this issue in the literature. For example, Leonard and Fiorelli [4] viewed reference points as virtual leaders to manipulate the geometry of autonomous vehicle group and direct the motion of the group.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 604–613, 2005.

In this paper, we investigate the collective behavior of multi-agent systems in high-dimensional space with point mass dynamics and with dynamic topology. We view the external control signal (or "mission") as virtual leader to direct, herd and/or manipulate the agent group behavior. During the course of motion, each agent is influenced by the external signal and the motion of other agents in the group. In order to generate the desired stable flocking motion, we introduce a set of switching control laws such that each agent regulates its position and velocity based on the desired velocity and the state information of its "neighbors". By using the control laws, all agent velocities asymptotically approach the desired value, collisions are avoided between the agents, and the final tight formation minimizes all agent potentials. One salient feature of this paper is that the self-organized global behavior is achieved via local feedback, i.e., the desired emergent dynamics is produced through local interactions and information exchange between the dynamic agents.

This paper is organized as follows: In Section 2, we formulate the problem to be investigated. By using some specific control laws, we analyze the system stability and the motion of the center of mass (CoM) in Section 3. Some numerical simulations are presented in Section 4. Finally, we briefly summarize our results in Section 5.

2 Problem Formulation

We consider a group of N agents moving in an n-dimensional Euclidean space, each has point mass dynamics described by

$$\begin{aligned} \dot{x}^i &= v^i \\ m_i \dot{v}^i &= u^i, \quad i = 1, \cdots, N \end{aligned} \tag{1}$$

where $x^i \in \mathbb{R}^n$ is the position vector of agent i, $v^i \in \mathbb{R}^n$ is its velocity vector, $m_i > 0$ is its mass, and $u^i \in \mathbb{R}^n$ is the (force) control input acting on agent i. $x^{ij} = x^i - x^j$ denotes the relative position vector between agents i and j.

Our objective is to make the entire group move at a desired velocity and maintain constant distances between the agents. In what follows, we will investigate the motion of the agent group in two different cases, that is, we consider the group motion in ideal case (i.e., velocity damping is ignored) and nonideal case (i.e., velocity damping is taken into account), respectively. For the two different cases, we propose two different control laws to achieve our control objective.

We first consider the ideal case. In this case, we try to regulate each agent velocity to the desired velocity, reduce the velocity differences between neighboring agents, and at the same time, regulate their distances such that their potentials become minimum. Hence, we choose the control law u^i for agent i to be

$$u^i = \alpha^i + \beta^i + \gamma^i \tag{2}$$

where α^i is used to regulate the potentials among agents, β^i is used to regulate the velocity of agent i to the weighted average of its neighbors, and γ^i is used

to regulate the momentum of agent i to the desired final momentum (all to be designed later). α^i is derived from the social potential fields which is described by artificial social potential function V^i, a function of the relative distances between agent i and its flockmates. Collision-free and cohesion in the group can be guaranteed by this term. β^i reflects the alignment or velocity matching with neighbors among agents. γ^i is designed to regulate the momentum among agents based on the external signal (the desired velocity). In some cases, by using such a of momentum regulation, we can obtain the explicit convergence rate of the CoM of the system. Note that, in fact, the design of α^i and β^i indicates that, during the course of motion, agent i is influenced only by its "neighbors", whereas γ^i reflects the influence of the external signal on the agent motion.

Certainly, in some cases, the velocity damping can not be ignored. For example, objects moving in viscous environment and mobile objects with high speeds such as supersonic aerial vehicles, are subjected to the influence of velocity damping. Then, in this case, the model in (1) should be in the following form

$$\begin{aligned} \dot{x}^i &= v^i \\ m_i \dot{v}^i &= u^i - k_i v^i \end{aligned} \tag{3}$$

where $k_i > 0$ is the "velocity damping gain", $-k_i v^i$ is the velocity damping term, and u^i is the control input for agent i. Here we assume that the damping force is in proportion to the magnitude of velocity and the damping gains k_i, $i = 1, \cdots, N$ are not equal to each other. In order to achieve our objective, we need to compensate for the velocity damping. Hence, we modify the control law u^i to be

$$u^i = \alpha^i + \beta^i + \gamma^i + k_i v^i. \tag{4}$$

3 Main Results

In this section, we investigate the stability properties of multiple mobile agents with point mass dynamics described in (1). We will present explicit control input in (2) for the terms α^i, β^i and γ^i. We will employ algebraic graph theory, differential inclusion, and nonsmooth analysis as basic tools for our discussion. Some concepts and results can be found in [8]–[12].

Throughout this paper, we assume that each agent is equipped with an onboard neighboring sensor which is used to sense the position and velocity information of its "neighbors" and an onboard signal detector which is used to detect the external signal, and assume that all sensors can sense instantaneously.

Following [2], we make the following definitions and assumptions.

Definition 1. *[2] (Neighboring graph) The neighboring graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, is an undirected graph consisting of a set of vertices, $\mathcal{V} = \{n_1, \cdots, n_N\}$, indexed by the agents in the group, and a set of edges, $\mathcal{E} = \{(n_i, n_j) \in \mathcal{V} \times \mathcal{V} \mid n_j \sim n_i\}$, containing unordered pairs of vertices that represent neighboring relations.*

The neighboring graph $\mathcal{G}$ is used to describe the sensor information flow in the group. In $\mathcal{G}$, an edge (n_i, n_j) means that agent i can sense agent j, and it

will regulate its state based on the position and velocity information of agent j. In this paper, we consider a group of mobile agents with dynamic topology. Let $\mathcal{I} = \{1, 2, \cdots, N\}$. Denote the set $\mathcal{N}_i \triangleq \{j \mid \|x^{ij}\| \leq R\} \subseteq \mathcal{I} \backslash \{i\}$ which contains all neighbors of agent i, where R is a positive constant and can be viewed as the agent sensing radius. This means that agent i can only obtain the state information of the agents who are contained in $\mathcal{N}_i$. Note that we assume the same sensing radius for all agents in the group. During the course of motion, the relative distances between agents vary with time, so the neighbors of each agent are not fixed, which generates the switching neighboring graph. Throughout this paper, we assume that the neighboring graph $\mathcal{G}$ remains connected, which ensures that the group will not be divided into several isolated subgroups. In order to depict the potential between two agents, we present the following definition.

Definition 2. *[2] (Potential function) Potential V^{ij} is a nonnegative function of the distance $\|x^{ij}\|$ between agents i and j, such that $V^{ij}(\|x^{ij}\|) \to \infty$ as $\|x^{ij}\| \to 0$, V^{ij} attains its unique minimum when agents i and j are located at a desired distance, and V^{ij} is increasing near $\|x^{ij}\| = R$.*

Functions V^{ij}, $i, j = 1, \cdots, N$ are the artificial social potential functions that govern the interindividual interactions. Function V^{ij} can be nonsmooth at $\|x^{ij}\| = R$, and in order to capture the fact that the motion of each agent only depends on the state information of its neighbors, we assume that potential $V^{ij}(\|x^{ij}\|)$ is a constant $V_R^{ij} = V^{ij}(R)$ for $\|x^{ij}\| > R$. One example of such potential functions is the following

$$V^{ij}(\|x^{ij}\|) = \begin{cases} a \ln \|x^{ij}\|^2 + \frac{b}{\|x^{ij}\|^2} & \text{for } 0 < \|x^{ij}\| \leq R \\ a \ln R^2 + \frac{b}{R^2} & \text{for } x > R \end{cases} \tag{5}$$

where a, b and R are some positive constants such that $R > \sqrt{b/a}$. Note that the assumption $R > \sqrt{b/a}$ is reasonable as this implies that the desired distance between the agents is smaller than the agent's sensing range R. It is easy to see that V^{ij} attains its unique minimum $a(1 + \ln(b/a))$ when $\|x^{ij}\| = \sqrt{b/a}$.

By the definition of V^{ij}, the total potential of agent i can be expressed as

$$V^i \triangleq \sum_{j \notin \mathcal{N}_i, j \neq i} V_R^{ij} + \sum_{j \in \mathcal{N}_i} V^{ij}\left(\left\|x^{ij}\right\|\right).$$

During the course of motion, each agent regulates its position and velocity based on the external signal and the state information of its neighbors. However, it is known that, in reality, because of the influence of some external factors, the reference signal is not always detected by all agents in the group. In this paper, we will consider the case that the signal is sent continuously and at any time, there exists at least one agent in the group who can detect it. In what follows, we only present the detailed analysis for the ideal case, since for the nonideal case, we only need to add the terms $k_i v^i$ $(i = 1, \cdots, N)$ to cancel the velocity damping. We take the control law u^i to be

$$u^i = -\sum_{j \in \mathcal{N}_i} w_{ij}\left(v^i - v^j\right) - \sum_{j \in \mathcal{N}_i} \nabla_{x^i} V^{ij} - h_s^i m_i\left(v^i - v^0\right) \tag{6}$$

where $v^0 \in \mathbb{R}^n$ is the desired common velocity and is a constant vector, h_s^i equals 1 if agent i can detect the reference signal and is 0 otherwise, $w_{ij} \geq 0$ and $w_{ii} = 0$, $i, j = 1, \cdots, N$ represent the interaction coefficients. In this paper, we assume that, if agents i and j are the neighboring agents, the interaction coefficient $w_{ij} = c_{ij}$ is fixed, where $c_{ij} > 0$ $(\forall\ i \neq j)$ is a constant, and otherwise $w_{ij} = 0$. This means that there is only two choices for the interaction coefficient between agents i and j that are c_{ij} and 0. Here we always assume that $c_{ij} = c_{ji}$, which means that the interaction between agents is reciprocal. We denote $W_\sigma = [w_{ij}]_\sigma \in \mathbb{R}^{N\times N}$ as the interaction coefficient matrix (coupling matrix), where σ is a switching signal and is a piecewise constant function $\sigma(t) : [0, \infty) \to \mathcal{P}$, $\mathcal{P}$ is a finite index set where the number of the indices is equal to the number of the connected neighboring graphs in the group. The switching signal σ relies on the distances between agents. Hence, W_σ is always symmetric, and by the assumption of the connectivity of the neighboring graph, W_σ is always irreducible.

3.1 Stability Analysis

Lemma 1. *[2] Function V^{ij} is regular everywhere in its domain. Moreover, the generalized gradient of V^{ij} at R and the (partial) generalized gradient of V^{ij} with respect to x^i at R are empty sets.*

Theorem 1. *By taking the control law in (6), all agent velocities in group (1) asymptotically approach the desired common velocity, collisions are avoided between the agents, and the group final configuration minimizes all agent potentials.*

This theorem becomes apparently true after Theorem 2 is proved, so we proceed to present Theorem 2.

We define the error vectors: $e_p^i = x^i - v^0 t$, and $e_v^i = v^i - v^0$, where t is time variable and v^0 is the desired common velocity. Then e_v^i represents the velocity difference vector between the actual velocity and the desired velocity of agent i. It is easy to see that $\dot{e}_p^i = e_v^i$ and $\dot{e}_v^i = \dot{v}^i$. Thus the error dynamics is given by

$$\begin{aligned} \dot{e}_p^i &= e_v^i \\ \dot{e}_v^i &= \tfrac{1}{m_i} u^i, \quad i = 1, \cdots, N. \end{aligned} \tag{7}$$

By the definition of V^{ij}, it follows that $V^{ij}(\|x^{ij}\|) = V^{ij}(\|e_p^{ij}\|) \triangleq \widetilde{V}^{ij}$, where $e_p^{ij} \triangleq e_p^i - e_p^j$, and hence $\widetilde{V}^i = V^i$ and $\nabla_{e_p^i} \widetilde{V}^{ij} = \nabla_{x^i} V^{ij}$. Thus, the control input for agent i in the error system has the following form

$$u^i = -\sum_{j\in\mathcal{N}_i} w_{ij} \left(e_v^i - e_v^j\right) - \sum_{j\in\mathcal{N}_i} \nabla_{e_p^i} \widetilde{V}^{ij} - h_s^i m_i e_v^i. \tag{8}$$

Theorem 2. *By taking the control law in (8), all agent velocities in the system described in (7) asymptotically approach zero, collisions are avoided between the agents, and the group final configuration minimizes all agent potentials.*

Proof. Consider the positive semi-definite function: $J = \frac{1}{2}\sum_{i=1}^{N}(\widetilde{V}^i + m_i e_v^{iT} e_v^i)$. It is easy to see that J is the sum of the total artificial potential energy and the total kinetic energy of all agents in the error system. Function J is continuous but may be nonsmooth whenever $\|e_p^{ij}\| = R$ for some $(i,j) \in \mathcal{I} \times \mathcal{I}$. Define the level set of J in the space of agent velocities and relative distances in the error system: $\Omega = \{(e_v^i, e_p^{ij}) \mid J \leq c,\ c > 0\}$. Though the neighboring relations vary with time, under the assumption of the connectivity of the neighboring graph $\mathcal{G}_\sigma$, the set Ω is compact. This is because the set (e_v^i, e_p^{ij}) such that $J \leq c$ is closed by continuity. Moreover, boundedness can be proved by connectivity. In fact, because the neighboring graph is always connected, there must be a path connecting any two agents i and j in the group and its length does not exceed $N-1$, and on the other hand, the distance between two interconnected agents is not more than R, hence, we have $\|e_p^{ij}\| \leq (N-1)R$. By similar analysis, we have $e_v^{iT} e_v^i \leq 2c/m_i$, thus $\|e_v^i\| \leq \sqrt{2c/m_i}$. Note that the restriction of J in Ω ensures collision avoidance and the differentiability of $\|e_p^{ij}\|$, $\forall\ i,j \in \mathcal{I}$.

By the definition of $\widetilde{V}^{ij}$, $\widetilde{V}^{ij}$ is continuous and locally Lipschitz. From Lemma 1, $\widetilde{V}^{ij}$ is regular everywhere in its domain and then $\widetilde{V}^i$ is regular everywhere, hence, J is regular as a sum of regular functions [12]. Then, we have

$$\partial J \subset \left[\sum_{j=2}^{N}\left(\partial_{e_p^1}\widetilde{V}^{1j}\right)^T, \cdots, \sum_{j=1}^{N-1}\left(\partial_{e_p^N}\widetilde{V}^{Nj}\right)^T, m_1 e_v^{1T}, \cdots, m_N e_v^{NT}\right]^T.$$

Hence, the generalized time derivative of J is

$$\dot{\widetilde{J}} \subset \sum_{i=1}^{N}\left(\bigcap_{\xi_i}\xi_i^T e_v^i\right) - e_v^T K\left[\left(L_\sigma \otimes I_n\right) e_v + \left(\cdots, \left(\nabla_{e_p^i}\widetilde{V}^i\right)^T, \cdots\right)^T + \left(H_s \otimes I_n\right) e_v\right]$$

where $\xi_i \in \sum_{j=1, j\neq i}^{N} \partial_{e_p^i}\widetilde{V}^{ij}$, $e_v = \left(e_v^{1T}, \cdots, e_v^{NT}\right)^T$ is the stack vector of all agent velocity vectors in the error system, $L_\sigma = [l_{ij}]_\sigma$ with $l_{ij} = -w_{ij}$ for all $i \neq j$ and $l_{ij} = \sum_{k=1, k\neq i}^{N} w_{ik}$ for all $i = j$, $\otimes$ stands for the Kronecker product, I_n is the identity matrix of order n, $\nabla_{e_p^i}\widetilde{V}^i = \sum_{j\in\mathcal{N}_i}\nabla_{e_p^i}\widetilde{V}^{ij}$, and $H_s = \mathrm{diag}\left(h_s^1 m_1, \cdots, h_s^N m_N\right)$. Due to the switching topology of the neighboring relations, L_σ and $\nabla_{e_p^i}\widetilde{V}^i$ are switching over time. By Lemma 1, we get $\dot{\widetilde{J}} \subset -\overline{co}\left\{e_v^T\left(L_\sigma \otimes I_n\right) e_v\right\} - \overline{co}\left\{e_v^T\left(H_s \otimes I_n\right) e_v\right\}$.

By matrix theory [8] and by the definition of matrix L_σ, it is easy to see that L_σ is positive semi-definite. On the other hand, by the connectivity of $\mathcal{G}_\sigma$, it follows that L_σ is irreducible and the eigenvector associated with the single zero eigenvalue is $\mathbf{1}_N = [1, \cdots, 1]^T \in \mathbb{R}^N$. From the proof of Theorem 1 in [3], we obtain that, for any graph $\mathcal{G}_\sigma$, the eigenvalues of $L_\sigma \otimes I_n$ are nonnegative, $\lambda = 0$ is an eigenvalue of multiplicity n, and $-\overline{co}\{e_v^T(L_\sigma \otimes I_n)e_v\}$ is an interval of the form $[l_1, 0]$ with $l_1 < 0$, and only when $e_v^1 = \cdots = e_v^N$, 0 is contained in $-\overline{co}\{e_v^T(L_\sigma \otimes I_n)e_v\}$. Furthermore, it is easy to see that matrix H_s is positive semi-definite, hence $-\overline{co}\{e_v^T(H_s \otimes I_n)e_v\}$ is an interval of form $[l_2, 0]$ with $l_2 < 0$,

and 0 is contained in $-\overline{co}\{e_v^T(H_s \otimes I_n)e_v\}$ only when $e_v^i = \mathbf{0}$ for each agent i with $h_s^i = 1$. Therefore, for any $z \in \dot{\widetilde{J}}$, $z \leq 0$, and only when $e_v^1 = \cdots = e_v^N = \mathbf{0}$, 0 is contained in $-\overline{co}\{e_v^T(L_\sigma \otimes I_n)e_v\} - \overline{co}\{e_v^T(H_s \otimes I_n)e_v\}$. This occurs only when $v^1 = \cdots = v^N = v^0$. It follows that $\dot{e}_v^i = \mathbf{0}$, and $\dot{e}_p^{ij} = \mathbf{0}$, $\forall\ (i,j) \in \mathcal{I} \times \mathcal{I}$. We use nonsmooth version LaSalle's invariance principle [10] to establish convergence of the system trajectories to the largest invariant subset of the set defined by $S = \{e_v \in \Omega \mid 0 \in \dot{\widetilde{J}}\}$. In the set, the agent velocity dynamics is $\dot{e}_v^i = -\frac{1}{m_i}\sum_{j\in\mathcal{N}_i} \nabla_{e_p^i}\widetilde{V}^{ij} = -\frac{1}{m_i}\nabla_{e_p^i}\widetilde{V}^i$. Thus, in steady state, all agent velocities in the error system no longer change and equal zero, and moreover, the potential $\widetilde{V}^i$ of each agent i is minimized. Collision-free can be ensured between the agents since otherwise it will result in $\widetilde{V}^i \to \infty$. □

From the proof of Theorem 2, it follows that, in steady state, all agent actual velocities no longer change and are equal to the desired velocity.

Remark 1. Note that, when all nonzero interaction coefficients equal 1, i.e., $c_{ij} = 1$, $\forall\ i \neq j$, $i, j \in \mathcal{I}$, the desired stable flocking can still be obtained, and here L_σ is the Laplacian matrix of $\mathcal{G}_\sigma$. But due to the differences between agents, we assume that the coupling coefficients between them are not equal to each other.

3.2 The Motion of the CoM

In what follows, we will prove that, when the external signal can always be detected by all agents, the velocity of the CoM can be estimated explicitly.

In this case, the control law in (6) has the following form

$$u^i = -\sum_{j\in\mathcal{N}_i} w_{ij}\left(v^i - v^j\right) - \sum_{j\in\mathcal{N}_i} \nabla_{x^i} V^{ij} - m_i\left(v^i - v^0\right). \tag{9}$$

The position vector of the CoM of system (1) is defined as $x^* = (\sum_{i=1}^N m_i x^i)/(\sum_{i=1}^N m_i)$. Thus, the velocity vector of the CoM is $v^* = (\sum_{i=1}^N m_i v^i)/(\sum_{i=1}^N m_i)$. By using control law (9) and by the symmetry of matrix W_σ and the symmetry of function V^{ij} with respect to x^{ij}, we get $\dot{v}^* = -v^* + v^0$. Suppose the initial time $t_0 = 0$, and $v^*(0) = v_0^*$. We get $v^* = v^0 + (v_0^* - v^0)e^{-t}$. Thus, it follows that, if $v_0^* = v^0$, then the velocity of the CoM is invariant and equals v^0 for all the time; if $v_0^* \neq v^0$, then the velocity of the CoM exponentially converges to the desired velocity v^0 with a speed of 1. Therefore, from the analysis above, we have the following theorem.

Theorem 3. *By taking the control law in (9), if the initial velocity of the CoM is equal to the desired velocity, then it is invariant for all the time; otherwise it will exponentially converge to the desired velocity.*

Remark 2. By the calculation above, we can see that, when the external signal can always be detected by all agents in the group, the velocity variation of the CoM does not rely on the neighboring relations or the magnitudes of the interaction coefficients. Even if the neighboring graph is not connected, the velocity

of the CoM still equals the desired velocity or exponentially converges to it, and the final velocities of all connected agent subgroups equal the desired velocity as well. However, in this case, the distance between two disconnected subgroups might be very far. Furthermore, it is obvious that the convergence rate of the system is slower than that of the CoM, hence, by using the control law in (9), if the initial velocity of the CoM is not equal to the desired velocity, then the fastest convergence rate of the system does not exceed the exponential convergence rate with convergence exponent 1.

3.3 Discussions on Various Control Laws

In the sections above, we introduced a set of switching control laws that enable the group to generate the desired stable flocking motion. However, it should be clear that control law (6) is not the unique control law to produce the desired motion for the agent group. In what follows, we will propose some other useful control laws to achieve our control objective.

Suppose that α^i and β^i rely on agent i's mass. The control law acting on agent i has the following form

$$u^i = -\sum_{j\in\mathcal{N}_i} m_i w_{ij}\left(v^i - v^j\right) - \sum_{j\in\mathcal{N}_i} m_i \nabla_{x^i} V^{ij} - h_s^i m_i\left(v^i - v^0\right). \tag{10}$$

In this case, for the error system (7), we choose the Lyapunov function $J = \frac{1}{2}\sum_{i=1}^{N}(\widetilde{V}^i + e_v^{iT} e_v^i)$. Following the analysis method in Theorem 2, we can show that the desired stable flocking motion will be achieved.

Definition 3. *Define the center of the system of agents as* $\overline{x} = (\sum_{i=1}^{N} x^i)/N$. *The average velocity of all agents is defined as* $\overline{v} = (\sum_{i=1}^{N} v^i)/N$.

It is obvious that the velocity of the system center is just the average velocity of all agents. When the external reference signal can always be detected by all agents in the group, i.e., $h_s^i \equiv 1$ for all $i \in \mathcal{I}$, by using the control law in (10), we have $\dot{\overline{v}} = -\overline{v} + v^0$. Suppose the initial time $t_0 = 0$ and $\overline{v}(0) = \overline{v}_0$. We get $\overline{v} = v^0 + (\overline{v}_0 - v^0)e^{-t}$. It is obvious that, if $\overline{v}_0 = v^0$, then the velocity of the system center is equal to the desired velocity v^0 for all the time, and if $\overline{v}_0 \neq v^0$, then the velocity of the system center exponentially converges to the desired velocity with a speed of 1.

4 Simulations

In this section, we will present some numerical simulations for system (1) in order to illustrate the theoretic results obtained in the previous sections.

These simulations are performed with ten agents moving in the plane whose initial positions, velocities and the neighboring relations are set randomly, but they satisfy: 1) all initial positions are chosen within a ball of radius $R^* = 10$[m] centered at the origin, 2) all initial velocities are set with arbitrary directions

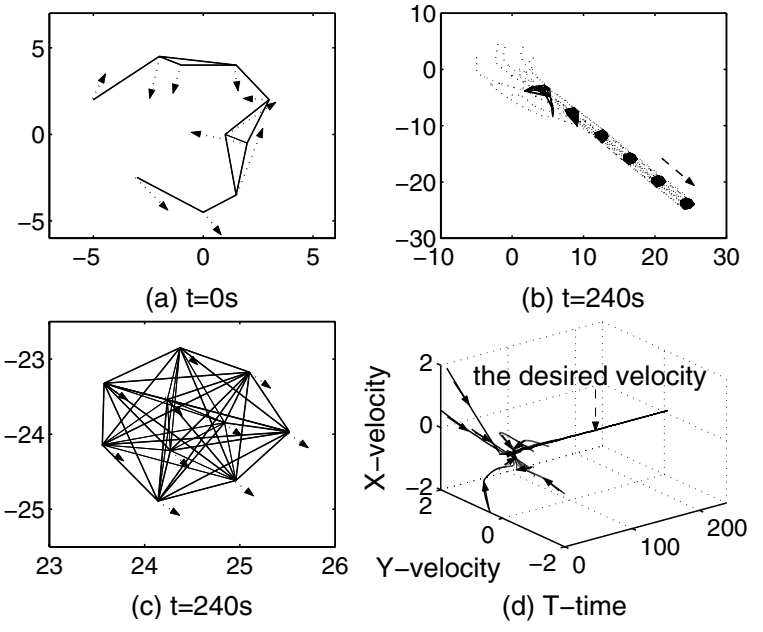


Fig. 1. The group state

and magnitudes in the range of (0, 4)[m/s], and 3) the neighboring graph remains connected. All agents have different masses to each other and they are set randomly in the range of (0, 1)[kg]. The external reference signal is sent continuously, and at any time, there exists only one agent who can detect it.

Fig. 1 shows the results in one of our simulations, where the control laws are taken in the form of (6) with the explicit potential function (5), where $a = b = 0.05$ and $R = 4$[m]. The interaction coefficient c_{ij} is generated randomly such that $0 < c_{ij} = c_{ji} < 1, \forall\, i \neq j,\, i, j = 1, \cdots, 10$. Correspondingly, the coupling matrix W_σ is generated such that the nonzero w_{ij} equals c_{ij}. We run the simulation for 240 seconds. Here we choose the desired common velocity to be $[0.1, -0.1]^T$, and the initial velocity of the CoM is $[0.1962, 0.0699]^T$. Fig. 1 (a) presents the group initial state, Fig. 1 (b) depicts the motion trajectories of all agents and six configurations of the group at different times, and Fig. 1 (c) shows the final configuration and velocity of the agent group (i.e., the desired common velocity $v^0 = [0.1, -0.1]^T$), where the solid lines represent the neighboring relations, the dotted lines represent the agent trajectories, the dotted lines with arrows represent the agent velocities, and the dashed line with arrow represents the motion direction of the group. Fig. 1 (d) is the velocity plot, where the solid arrow indicates the tendency of velocity variation, and it distinctly demonstrates that all agent velocities asymptotically approach the desired velocity.

Hence, numerical simulations also indicate that, by using the control law in (6), the desired stable flocking motion can be achieved.

5 Conclusions

We have investigated the collective behavior of multiple dynamic agents moving in an n-dimensional Euclidean space with point mass dynamics and with dynamic topology, and presented some switching control laws which ensure the group to generate the desired stable flocking motion. The control laws are a combination of attractive/repulsive and alignment forces, and they ensure that all agent velocities asymptotically approach the desired velocity, collisions are avoided between the agents, and the final tight formation minimizes all agent potentials. Moreover, when the external reference signal can always be detected by all agents, the velocity of the CoM either is equal to the desired velocity or exponentially converges to it. Furthermore, when the velocity damping is taken into account, we can properly modify the control laws to generate the desired stable flocking motion. Finally, we provided some numerical simulations to further illustrate our theoretical results.

Acknowledgement. This work was supported by National 973 Program (No. 2002CB 312200), NSFC (No. 10372002 and No. 60274001), and Engineering Research Institute of Peking University. Corresponding author: Professor Long Wang.

References

1. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics (ACM SIGGRAPH '87 Conf. Proc.) **21** (1987) 25–34
2. Tanner, H.G., Jadbabaie, A., Pappas, G.J.: Stable flocking of mobile agents, Part I: Fixed topology; Part II: Dynamic topology. Proc. IEEE Conf. Decision Contr. **2** (2003) 2010–2015; 2016–2021
3. Shi, H., Wang, L., Chu, T., Zhang, W.: Coordination of a group of mobile autonomous agents. Proc. Int. Conf. Adv. Intell. Syst. Theor. Appl. Luxembourg, Nov. (2004)
4. Leonard, N.E., Fiorelli, E.: Virtual leaders, artificial potentials and coordinated control of groups. Proc. IEEE Conf. Decision Contr. **3** (2001) 2968–2973
5. Wang, L., Shi, H., Chu, T., Zhang, W., Zhang, L.: Aggregation of forging swarms. Lecture Notes in Artificial Intelligence, Springer-Verlag **3339** (2004) 766–777
6. Shi, H., Wang, L., Chu, T.: Swarming behavior of multi-agent systems. J. Control Theory and Applications **2** (2004) 313–318
7. Mu, S., Chu, T., Wang, L.: Coordinated collective motion in a motile particle group with a leader. Physica A **351** (2005) 211–226
8. Horn, R.A., Johnson, C.R.: Matrix Analysis. New York: Cambridge Univ. Press (1985)
9. Godsil, C., Royle, G.: Algebraic Graph Theory. New York: Springer–Verlag (2001)
10. Shevitz, D., Paden, B.: Lyapunov stability theory of nonsmooth systems. IEEE Trans. Automat. Contr **39** (1994) 1910–1914
11. Paden, B., Sastry, S.: A calculus for computing Filippov's differential inclusion with application to the variable structure control of robot manipulators. IEEE Trans. Circuits Syst. **34** (1987) 73–82
12. Clarke, F.H.: Optimization and Nonsmooth Analysis. New York: Wiley (1983)

Cultural and Biological Evolution of Phonemic Speech

Bart de Boer

Kunstmatige Intelligentie, Rijksuniversiteit Groningen, Grote Kruisstraat 2/1,
9712 TS Groningen, The Netherlands
b.de.boer@ai.rug.nl
http://www.ai.rug.nl/~bart

Abstract. This paper investigates the interaction between cultural evolution and biological evolution in the emergence of phonemic coding in speech. It is observed that our nearest relatives, the primates, use holistic utterances, whereas humans use phonemic utterances. It can therefore be argued that our last common ancestor used holistic utterances and that these must have evolved into phonemic utterances. This involves co-evolution between a repertoire of speech sounds and adaptations to using phonemic speech. The culturally transmitted system of speech sounds influences the fitness of the agents and could conceivably block the transition from holistic to phonemic speech. This paper investigates this transition using a computer model in which agents that can either use holistic or phonemic utterances co-evolve with a lexicon of words. The lexicon is adapted by the speakers to conform to their preferences. It is shown that although the dynamics of the transition are changed, the population still ends up of agents that use phonemic speech.

1 Introduction

All spoken human languages are phonemically coded, that is, they have a large repertoire of words that are built up of a far smaller number of basic building blocks[1]. Thus the words "tea" and "eat" have different meanings, even though they are made up of the same basic sounds. The repertoires of calls of higher primates, on the other hand, are not phonemic. Although their calls are sometimes made up of smaller units such as in the long calls of gibbons, [3] it is not likely that the order of the units influences the meaning of the calls, or that new calls can be created by rearranging the units. Such systems are called *holistic* in this paper.

As our closest evolutionary relatives (Bonobos, Chimpanzees, Gorillas, Orangutans) all use holistic call systems, it can be safely assumed that our last common ancestor also used a holistic call system. At some point in evolution the call system must have made the transition from a holistic to phonemic. This paper addresses an aspect

[1] It is likely that humans use a combination of holistic and phonemic storage. Infants probably store the first words they learn very accurately, and analyze them into building blocks only later *e.g.* [1]. In adult language, too, there are some utterances that have communicative function and are learned, but that fall outside the phonology of the language (called "protosyllabic fossils" in [2]). Examples are utterances such as "pssst", "pfffff", and "tsk tsk". Such utterances are probably stored holistically.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 614 – 623, 2005.

of the question of how this transition can have taken place, and investigates it with the use of a computer model.

Phonemic call systems have a number of advantages over holistic call systems. As they make use of a limited number of discrete building blocks, utterances become more robust. Small errors in pronunciation of a building block will not immediately change it into another building block. This is an example of categorical perception, a phenomenon that is important in the perception of speech [4]. Also, phonemic systems can be made productive: new utterances can be formed by recombining the building blocks in novel ways. Finally, phonemic coding makes it possible to store large repertoires of utterances more compactly. Whereas holistic utterances need to be stored in complete detail, phonemic utterances can be stored in terms of strings of building blocks, while only the building blocks themselves need to be stored in detail. It is this aspect that this paper will focus on.

Although phonemic coding is advantageous for systems with a large number of utterances, holistic systems appear to be preferred for smaller numbers of utterances. This is understandable, as storage complexity and robustness are comparable for small systems, while new utterances can also be created easily in both types. The cognitive complexity of a phonemic system, however, is much higher. It is therefore understandable that a call system would evolve from holistic to phonemic as it grows in size.

This paper does not model the emergence of phonemic coding as such, as is done in for example [5], or the evolution of learning behavior [6], but focuses on the dynamics of the interaction between cultural and biological evolution. The influence of culture is important in the evolution of language *e. g.* [7, 8], but it could be seen as a complicating factor in the transition from a holistic to a phonemic sound system: languages adapt to the abilities of the language users [9] and it can therefore be assumed that a population of holistic learners will shape the language towards holism. As it is assumed that holism is a better strategy for small systems, the system that exists in the population will at first be optimized for holistic learners. When the system becomes bigger, it could become stuck in a state where phonemic coding would *in principle* be more optimal, but where the existing holistic system (as preferred and perpetuated by the holistic learners in the population) causes the fitness of phonemic learners to remain low. This paper investigates the interaction between cultural evolution of a repertoire of sounds and the "biological" evolution of the acquisition strategies in a population of language learners.

2 Phonemic and Holistic Acquisition

In a computer model that investigates the evolution of phonemic acquisition, there must be agents that can both learn a system of speech sounds as a set of holistic motor programs, or as a set of utterances that consist of smaller building blocks and that are thus phonemically coded. In order to implement this, accurate definitions of holistic and phonemic storage are required. In this paper, a holistic system uses only one level of storage. All utterances in the lexicon are stored as exact motor programs. A phonemic system, on the other hand, uses two levels of storage. These are the level of the lexicon and the level of the gestures that make up the building blocks out of which the lexicon is built up. In real language, these could be phonemes, syllables, gestures or

any other realistic primitive. The gestures that make up the building blocks are stored exactly, just as in the holistic system. The lexicon, however stores words as sequences of building blocks, without specifying the details of the building blocks. Learning and storing a pointer to a building block requires less space and effort than learning and storing the actual gestures of the building block. A holistic gesture must be stored with maximum accuracy, as a small change might result in a complete change in meaning. As fewer basic gestures are used in a phonemic system, the margin for error is greater, and storage and learning becomes easier. The two systems are illustrated in figure 1.

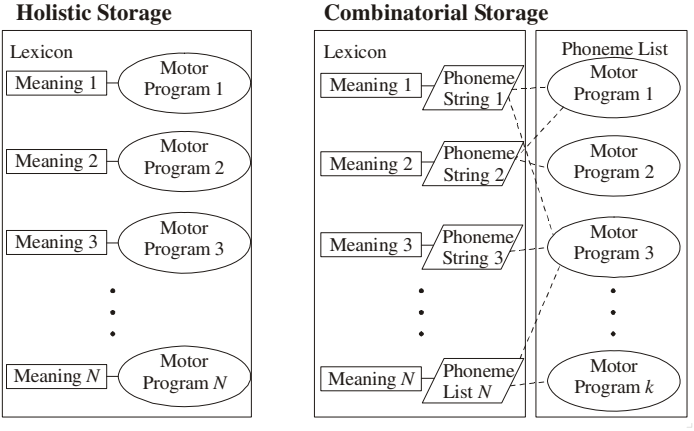


Fig. 1. Diagrams of holistic storage and phonemic storage

If the number of motor programs that are actually used in the lexicon is much smaller than the number of possible motor programs, it can be assumed that the storage of complete motor programs requires more space than storing symbols that represent them. In that case, the lexicon can be stored more compactly using phonemic storage than using holistic storage.

The fitness of agents that use holistic and phonemic coding is determined by a number of factors. One possible factor is the amount of storage required to store a given lexicon. Another factor is the robustness and the communicative success in noisy conditions that can be achieved *e. g.* [10]. A third factor is the amount of cognitive effort that is needed for learning, processing and producing the lexicon. In this paper only the storage requirement is taken into account.

The space s necessary for storing a lexicon L is as follows:

$$s = \begin{cases} \sum_{\forall w_i \in L} \alpha \cdot l_i & \text{, for a holistic system} \\ \alpha \cdot n + \sum_{\forall w_i \in L} \beta \cdot l_i + \gamma & \text{, for a phonemic system} \end{cases}, \tag{1}$$

where α is the number of bits needed to store all information about the motor program of a certain gesture, l_i is the number of gestures in word w_i, n is the number of differ-

ent articulatory gestures (building blocks or "phonemes") used by a phonemically coding agent, β is the number of bits needed to specify a phoneme in a word and γ the overhead needed for a phonemically coded system. From these equations it is clear that holistic coding is more efficient for a lexicon that has only words consisting of a single gesture. Phonemic coding is more efficient if there are many words re-using the same gestures in the lexicon. The exact parameter values define the transition point. It has not been attempted to estimate realistic values for the parameters. Our knowledge of how speech is stored in the brain is insufficient for this. Nevertheless, there are a few relations between the parameters that can be determined from first principles. It must be true that $\alpha \gg \beta$, as specifying a gesture exactly is more complex than referring to it. It must also be true that $\beta \equiv \log_2 n$. The more phonemes there are, the more bits are needed to distinguish them, and information theory [11] teaches that this number is proportional to the logarithm of the number of phonemes. Furthermore, it is impossible to use all potential articulatory gestures as distinctive speech sounds (phonemes). In order to preserve acoustic distinctiveness with a margin of error there must be unused acoustic and articulatory space between different gestures. This means that there are fewer possible phonemes than possible gestures. This can be formulated in the equation: $n_{\max} \ll 2^{\alpha}$, where $n_{\max}$ is the maximal number of phonemes, and 2^{α} is the maximum number of possible gestures, as this is the number of different strings of α bits (if there were more gestures, more bits would be needed to distinguish them).

The above considerations imply that a lexicon with a sufficiently large number of words always needs to contain words that consist of multiple gestures. Therefore, for sufficiently large lexicons, phonemic coding will be more efficient. If the lexicon is small, on the other hand, it can consist of single-gesture words without impeding distinctiveness. Therefore holistic coding can be more efficient.

At some intermediate size, a transition from optimality of holistic systems to optimality of phonemic systems must occur. At what size the transition will actually take place is not just determined by the parameters (which can be considered genetically determined factors) but also by what system is already in use in a population, i.e. cultural factors. Holistic learners will prefer a system with many different gestures and short words, while phonemic learners prefer a system with long words and few different gestures. Through self-organization, the sound system in the population will tend to adapt to the preferences of the majority of the population. This will have effects on the dynamics in a system where both the learned system of speech sounds and the learners themselves change.

3 The Model

A computer model has been implemented to investigate the dynamics of a system that combines genetic evolution of learners with cultural evolution of a repertoire of speech sounds. There are two kinds of agents in the model: holistic learners and phonemic learners. This is a simplification, because humans probably use both strategies, but this makes it much easier to understand the dynamics of the model. As there are only two types of agents, the population could have been modeled as a set of one-bit genomes. Instead, it has been decided to model only the fraction of agents that learn

holistically, p_h and the fraction that learns phonemically, p_p. Although this way of modeling a population is perhaps unusual in artificial life, where one generally prefers to model complete agents, it is an old tradition in theoretical biology [12]. Because there are only two agent types, it follows that $p_h + p_p = 1$. As all agents within each group are identical, fitness can be associated with the group instead of with the individuals. The fitness of the two groups is indicated with f_h and f_p, respectively and they are used to calculate the fractions of each type of agent in the next generation. There is also the probability μ of one type of agent mutating into the other type of agent (set to 0.1 in the simulations presented here). The equations for calculating the proportion of agents in the next generation are as follows:

$$\begin{aligned} p_{h,t+1} &\leftarrow \nu\left(f_{h,t} \cdot p_{h,t} + \mu \cdot f_{p,t} \cdot p_{p,t}\right) \\ p_{p,t+1} &\leftarrow \nu\left(f_{p,t} \cdot p_{p,t} + \mu \cdot f_{h,t} \cdot p_{h,t}\right) \end{aligned}, \tag{2}$$

where ν is a factor that causes $p_{h,t+1}$ and $p_{p,t+1}$ to sum to one.

All agents in the population, both holistic and phonemic learners, share the same lexicon. The fitness of each group is determined by the number of bits needed to store this lexicon. The lexicon consists of a list of words that each in turn consist of one or more basic gestures. These basic gestures can be represented by symbols, and phonemic learners use them as their building blocks (phonemes). Equation 1 is used for calculating the number of bits needed to store a repertoire. An example of a repertoire and the number of bits needed to store it is given in figure 2. The values of the parameters used to calculate these numbers are as follows: number of bits for a gesture (α): 10, penalty for using a phonemic system (γ): 30 bits, number of bits used per phoneme (β): $\log_2 n$. These same parameters were used in all the simulations as well. Note that it is possible to optimize the lexicon used in the example for both holistic and phonemic learners. For holistic learners, a system using the same gestures and having the same number of words, but needing only 110 bits of storage can be constructed (by changing *ao* into *o* and *aea* into *ea,* for example). Phonemic coding could also be substantially more efficient, for example by using two one-phoneme, four two-phoneme and two three-phoneme words. In that case only two different phonemes would be needed, resulting in a size of 66 bits. The resulting lexicons are shown on the right side of figure 2. It is clear that for lexicons of the same size, holistic and phonemic learners prefer very different words.

Given the number of bits s_p and s_h for phonemic and holistic learners, respectively the fitnesses are calculated as follows:

$$\begin{aligned} f_p &= 1 - \frac{s_p}{s_p + s_h} \\ f_h &= 1 - \frac{s_h}{s_p + s_h} \end{aligned}. \tag{3}$$

Note that fitness is relative: the fitness of one group of agents depends on the fitness of the other group. Thus the two groups *co-evolve*.

Each generation, the lexicon can be modified. A new word can be added with a certain probability and the words in the lexicon are modified in correspondence with the preferences of the agents in the population. The new word that is added is the

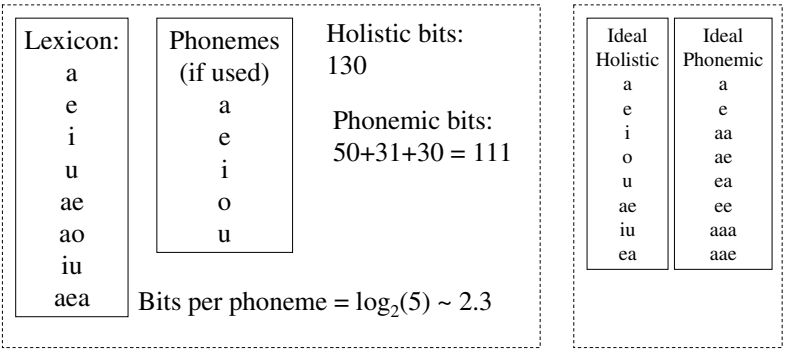


Fig. 2. Example of a lexicon and the number of bits needed for holistic and phonemic storage. For parameter values, see the text.

shortest word that is not already present. Addition is a holistic process, in the sense that new articulatory gestures can be created. It was decided to implement addition as a holistic process, because phonemic addition would not as readily add new phonemes to a growing repertoire, and because in the stage before transition, the majority of the population is expected to consist of holistic learners. A possible variant of the model would be to make the type of addition used depend on the proportion of holistic and phonemic agents in the population. As words can sometimes be removed from the lexicon the added word can be shorter than the longest word in the lexicon. Words are added with a probability of 10% per generation.

The lexicon can also be modified to better suit either type of agent. To suit holistic agents (who dislike long words) the longest word is removed from the lexicon and replaced with an unused shorter word, if possible. New articulatory gestures can be introduced as a side effect. To suit phonemic agents, first the phoneme that is least often used is found, and then the word in which it occurs most frequently. It is then attempted to replace this word with the shortest word that is build up of the phonemes already present in the lexicon. This can cause phonemes to disappear from the lexicon and average word length to increase. The first process puts pressure on the lexicon for shorter words and more phonemes, while the second process puts pressure on the lexicon for longer words and fewer phonemes.

If culture is turned on in the simulation presented here, two words can be modified per generation. No modification takes place if there is no culture. For each modification either a holistic or a phonemic agent is selected with probabilities that are proportional to their abundance in the population. Thus, if there are many holistic agents, the lexicon is pushed towards holism. If there are many phonemic agents, it is pushed towards phonemic coding.

4 Results

In the experiments, the population is initialized with 50% holistic and 50% phonemic agents. The lexicon is initialized with a single word (consisting of a single gesture). The maximum number of different gestures (n_{max}) is 16. The rest of the parameters are

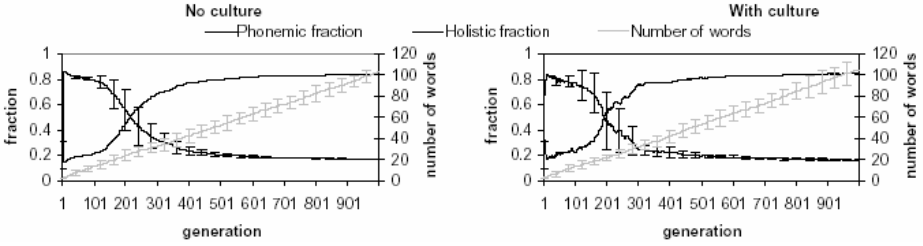


Fig. 3. Comparison of average behavior (over 10 runs) of a population without culture (left graph) and a population with culture (right graph). As the fractions of phonemic and holistic fractions are symmetrical, error bars showing standard deviation are only shown for the holistic fraction.

as described above. The result of running the model without and with cultural influences is shown in figure 3. As can be seen from this figure, the proportion of holistic agents rapidly rises in the beginning. Holistic agents have higher fitness for small lexicons than phonemic agents (as would be expected). The population then remains almost exclusively holistic for a while (a minority of phonemic agents remains present because of mutation). After a critical number of words is reached, the population makes a transition from a holistic majority to a phonemic majority.

At first sight, there seems to be little difference in average behavior between systems with culture and without culture. However, as can be observed in the graph, the error bars in the graph for populations with culture are much larger. Apparently there is a difference between the two cases, but it is masked by the averaging procedure.

The difference is caused by the fact that the transition is much faster for populations with culture than for populations without. This is illustrated in figure 4 which shows typical runs for systems without and with culture. For faster transitions, the standard deviation will be calculated over runs in which some of the populations are still in the holistic state and others in the phonemic state, hence increasing the standard deviation. But this only reflects the fact that the standard deviation is calculated over a distribution with two peaks, not that the peaks themselves are broader.

A comparison of the number of generations needed for the fraction of holistic agents to change from above 60% to below 40% confirms this. This is 52.3 generations ($\sigma = 14.6$) for the population without culture and 12.1 generations ($\sigma = 5.6$) for the population with culture. There appears to be no significant difference for the time at which the drop takes place; this happens after 191 ($\sigma = 42.6$) and 194 ($\sigma = 49.7$) generations, respectively. There is a remarkably large variation in the number of generations to the transition, but this is due to the fact that the increase of the number of words is a random process. The large variation disappears when one looks at the number of words at which the transition takes place. The number of words at which the drop starts (the 60% holistic learners threshold is crossed) is 22.4 ($\sigma = 0.52$) for populations without culture and 21.4 ($\sigma = 1.07$) for systems with culture. Although this is a significant difference, it is probably caused by the fact that populations with culture go through the transition faster than populations without culture and it probably does not reflect a real difference in the size of the lexicon at which transition starts. Systems probably start to transition when the number of words exceeds the

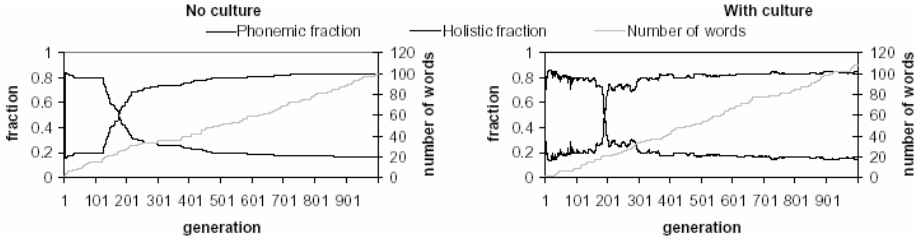


Fig. 4. Typical runs from a system without culture (left graph) and a system with culture (right graph). Note the faster transition in the graph with culture.

number of possible articulatory gestures. This is borne out by inspection of the data sets and confirmed for systems with 32 instead of 16 possible different gestures. The number of combinatorial agents starts to rise once complex utterances become necessary. Even systems that have been optimized for holistic learners then become more efficient for phonemic learners.

These results are robust for changes of parameters. As has been said above, changing the number of possible gestures does not affect the qualitative behavior. Changing the number of modifications that are made to the repertoire does not appear to affect the results. Changing the mutation rate to a lower value (0.03) causes agents of the unfavored type to become rarer. As it takes slightly longer for the number of agents to rise when the transition starts, this causes transitions to occur slightly later (at 24–25 words) An example is given in figure 5.

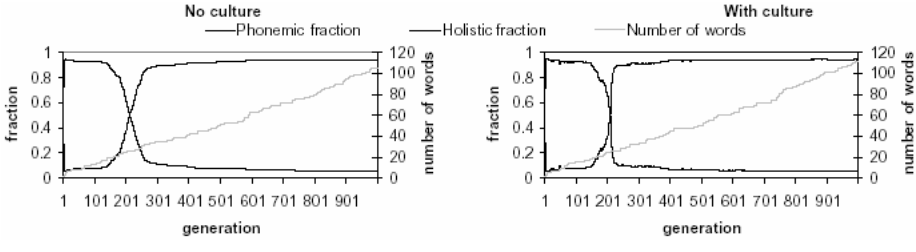


Fig. 5. Typical runs from a system with low mutation rate (m=0.03). Note the similarity with the transition in the previous figure.

5 Discussion and Conclusion

In the experiments presented here, holistic and phonemic learners had to compete. Their fitness depended on the amount of storage that was needed for storing a lexicon that evolved (culturally) together with the agents. Both types of agents changed the lexicon such that it would be easier for them to learn, causing the lexicon to tend towards optimality for the agents that had the majority in the population. It could be imagined that these cultural dynamics could cause the system to remain stuck in a local optimum, such that it would not make the transition from the (initially optimal) holistic lexicon to the

(ultimately optimal) phonemic lexicon. This was not observed in the experiments for different parameter settings. It can therefore be tentatively concluded that cultural inertia does not greatly influence the transition from holistic to phonemic speech.

The dynamics of a population with culture was observed to be different than that from a population without culture, however. It was observed that a population can change much more abruptly from holistic to phonemic language use if there is co-evolution of the culturally transmitted system of speech sounds with the genetically evolving language learners. This can be explained by the fact that once the proportion of phonemic learners starts to increase, the lexicon will also be changed to become more optimal for the phonemic agents. This increases the fitness of the phonemic agents, thus accelerating the transition. So although culture might at first be an obstacle to the transition, in the end it accelerates it.

These observations illustrate that the evolution of speech (and language) cannot be seen as either purely genetic or purely cultural evolution. Both mechanisms must be taken into account. They also confirm that phonemically coded systems win over holistically coded systems, at least as far as storage is concerned. They win, even though at first there is a cultural evolution towards systems that are more learnable for holistic learners.

The model that was used is admittedly simplistic, and only a limited number of experiments was performed. The model was made so simple in order to create a very basic model that nevertheless has interaction between the evolution of the acquisition of complex speech and the (cultural) evolution of the speech sounds themselves. There are many ways in which this work can be extended: a mathematical analysis of the dynamics would seem to be possible. Also, the influence of the different parameters can be investigated, it could be investigated at which point the transition from holistic to phonemic coding takes place exactly, different variants on the optimization procedures and the addition of new words could be tried out and many other small variants.

More interesting, however, is to strive for more realism in the simulation. As the model is about the evolution of acquisition, it is important to try to model a population of agents that really acquire the system of speech sounds. The acquisition mechanism should have parameters that make it exploit phonemic structure to a higher or lower degree, and these parameters should be able to evolve. The time it takes to acquire a repertoire of speech sounds and the accuracy with which this happens could be taken into account in the fitness function. A next step could then be to get rid of the global lexicon, and have them be emergent in the population, just as the sound systems are emergent in the population in [13, 14]. Also, more realistic constraints on production and perception could be added. Such a model would already be quite realistic, but also have much more complicated and hard-to-understand dynamics. The model as presented in this paper gives a basis for understanding such dynamics.

Another important research topic would be finding independent evidence with which to compare the results of such a computer simulation. This can be evidence from language acquisition, evidence from the fossil record, or evidence from animal call systems. Perhaps the study of call systems from closely related primate species (the different species of gibbon, [15] for example) can provide insight into the circumstances under which a holistic call system can change into a phonemic call system.

The understanding of the dynamics of the evolution of speech is a crucial piece of the puzzle that cannot be found either by studying animals or by studying the fossil record. The interaction between the evolution of a cultural repertoire of speech sounds and the physical adaptations for processing, producing and perceiving them cause the evolution of speech to have very complex dynamics. Computer models can provide insights in these dynamics. The computer model presented here is an attempt to provide insight in the interaction between cultural and genetic evolution, and it is hoped that it can be used as an inspiration for further research.

References

1. Houston, D. M. & Jusczyk, P. W.: The role of talker-specific information in word segmentation by infants. Journal of Experimental Psychology: Human Perception and Performance 26: (2000) 1570–1582
2. Jackendoff, R.: Foundations of language. Oxford University Press, Oxford (2002)
3. Mitani, J. C. & Marler, P.: A phonological analysis of male gibbon singing behavior. Behaviour 109: (1989) 20–45
4. Cooper, F. S., Delattre, P. C., Liberman, A. M., Borst, J. M. & Gerstman, L. J.: Some Experiments on the Perception of Synthetic Speech Sounds. Journal of the Acoustical Society of America 24: (1952) 597–606
5. Oudeyer, P.-Y.: Phonemic coding might be a result of sensory-motor coupling dynamics. In: Hallam, J. (ed.): Proceedings of the International conference on the simulation of adaptive behavior (SAB). MIT Press, Edinburgh (2002) 406–416
6. Hurford, J.: The evolution of the critical period for language acquisition. Cognition 40: (1991) 159–201
7. Steels, L.: Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation. In: Hurford, J. R., Michael, S.-K. & Knight, C. (eds.): Approaches to the Evolution of Language. Cambridge University Press, Cambridge (1998) 384–404
8. Smith, K., Kirby, S. & Brighton, H.: Iterated Learning: a framework for the emergence of language. Artificial Life 9: (2003) 371–386
9. Zuidema, W.: How the poverty of the stimulus solves the poverty of the stimulus. In: Becker, S., Thrun, S. & Obermayer, K. (eds.): Advances in Neural Information Processing Systems 15. MIT Press, Cambridge, MA (2003) 51–58
10. Nowak, M. A. & Krakauer, D.: The evolution of language. Proceedings of the National Academy of Sciences 96: (1999) 8028–8033
11. Shannon, C. E.: A mathematical theory of communication. The Bell system technical journal 27: (1948) 379–423,623–656
12. Maynard Smith, J.: Models in Ecology. Cambridge University Press, Cambridge (1974)
13. De Boer, B.: The origins of vowel systems. Oxford University Press, Oxford (2001)
14. De Boer, B.: Self organization in vowel systems. Journal of Phonetics 28: (2000) 441–465
15. Geissmann: Duet-splitting and the evolution of gibbon songs. Biological Reviews of the Cambridge Philosophical Society 77: (2002) 57–76

Grammar Structure and the Dynamics of Language Evolution

Yoosook Lee[1], Travis C. Collier[1], Gregory M. Kobele[2], Edward P. Stabler[2], and Charles E. Taylor[1]

[1] Dept. of Ecology and Evolutionary Biology, UCLA, Box 951606, Los Angeles, CA 90095-1606, USA
[2] Linguistics Dept., UCLA, Box 951543, Los Angeles, CA 90095-1542, USA
http://taylor0.biology.ucla.edu/al/

Abstract. The complexity, variation, and change of languages make evident the importance of representation and learning in the acquisition and evolution of language. For example, analytic studies of simple language in unstructured populations have shown complex dynamics, depending on the fidelity of language transmission. In this study we extend these analysis of evolutionary dynamics to include grammars inspired by the principles and parameters paradigm. In particular, the space of languages is structured so that some pairs of languages are more similar than others, and mutations tend to change languages to nearby variants. We found that coherence emerges with lower learning fidelity than predicted by earlier work with an unstructured language space.

1 Introduction

The evolutionary dynamics of language provides insight into the factors allowing subpopulations to converge on common or similar languages. The problem has a more general significance for robotics and artificial life as a clear and empirically supported platform for the study of how coherent behavior can emerge in a population of distributed adaptive agents.

Of particular interest from the perspective of evolutionary dynamics are insights into the means and value of conserving linguistic diversity. The practical importance of linguistic diversity has attracted some attention [23,21], though perhaps not as much as biological diversity. Recent studies that have applied a biological perspective to the evolution of linguistic convergence and diversity have shown promising results [7,8,22,17,11,20,15]. Most such studies that apply a biological perspective to language evolution have been based on very simple languages arbitrarily related to each one another. We believe these studies may be enriched by a more realistic description of language.

Language models based on the Chomskian paradigm [1,2] view language as an aspect of individual psychology. There has been some debate about the extent to which the underlying representation of languages are inherited or learned and how language impacts fitness. Pinker and Bloom, for example, suggest that a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 624–633, 2005.

language instinct constrained by universal grammar sets the stage for language acquisition which then contributes to individual fitness [19,18]. Hauser, Chomsky and Fitch argue more recently that while certain perceptual and articulatory abilities may have been selected for, it remains unclear how the most fundamental aspects of human language emerged [9,6]. All parties agree that linguistically relevant properties are to some extent learned through cultural transmission and change through time. How this might occur has been the subject of many analytic and simulation studies [15,16,20,11,22].

As an organism is determined in part by its genome, language is determined in part by a lexicon of generators which in turn determine its phonology, semantics, morphology and syntax; these properties may evolve [10,13]. Both the genome and at least part of language is inherited with variation, and therefore potentially a target for natural selection. These similarities have lead some researchers to adopt a quasi-species model [5,4] for describing the dynamics of language evolution [17,12]. In their model, grammars are mutationally equidistant from each other with arbitrarily assigned similarity. It seems, however, that the kind of changes language actually undergoes is much smaller than what this model seems to predict – the language of a child is, more or less, the same as that of its linguistic community. This suggests an approach where the similarity between languages is correlated with their distance from each other in mutational space.

In this paper, we study how certain properties of the space of possible languages and learning mechanisms impact language change. We introduce a regularity in the language space by viewing the locus of language transmission as a series of learned parameters and calculating the similarity between languages as the proportion of parameters that agree. We explore the effect of this simple regularity on the dynamics of language evolution primarily through simulations. These simulations go beyond previous analytic studies of simple models, and we find that structure has a significant impact on stability results.

2 Methods

Consider a fully-connected finite population of N individuals, each of whom possesses a language which is encoded as a sequence of l linguistic 'components' or 'parameters'. Each parameter can take only a limited number d of values. For

Table 1. Parameters used in the simulations

Symbol	Parameters	value(s)
N	population size	500
f_0	base fitness	10^{-3}
l	Number of language parameters	1, 2, 3, or 6
d	Number of values per each parameter	64, 8, 4, or 2
n	Number of possible grammars	$64(= d^l)$
	Number of time steps	100,000

example, a language L with 10 parameters each taking 3 values (A,B,C) can be represented by a linear sequence like `AABABCBABA`. (This string is not an example of a statement from the language, but rather a represents the language itself.) Such a representation is in the spirit of Chomsky's "principles and parameters" approach to language[15]. To allay a potential source of confusion: parameters, as we use them here, correspond to whatever the relevant differences are between language *at the level of description relevant to transmission.* This will correspond to parameters in the Chomskian sense just in case these latter parameters are appropriately relevant to linguistic transmission. We are throughout assuming that whatever is being transmitted can be usefully viewed (at least to a first approximation) as a finite sequence of finite-valued parameters.

Representing a language as a sequence, we define the language similarity between individual i and j, denoted a_{ij}, as the proportion of parameters on which the two individuals agree. For example, the language similarity between an individual i whose language is represented as AAA and an individual j whose language is represented as ABA is 2/3 and $a_{ij} = a_{ji}$.

The fitness of an individual has two parts: the base fitness, denoted f_0, and a *linguistic merit* proportional to the probability that the individual is able to successfully communicate with another, selected at random from his neighbors. The linguistic merit of an individual is proportional to the sum of language similarity between the individual and others it is in linguistic contact with (which is the entire population for this model). The overall fitness of an individual, f_i, is described as the following, as in [16]:

$$f_i = f_0 + \frac{1}{2}\sum_{j=1}^{N}(a_{ij} + a_{ji}) = f_0 + \sum_{j=1}^{N} a_{ij} \tag{1}$$

noting that $a_{ij} = a_{ji}$ according to our definition of similarity.

At each time step, an individual is chosen to reproduce randomly and independently with a probability according to relative fitness. Reproduction can be thought of either as the individual producing an offspring which inherits the parent's language and replaces another in the population, or another individual changing its language to match the "teacher's" language. We will use the former terminology.

The offspring learns the parent's language with a certain *learning fidelity*, q. This learning fidelity is properly a function of the specifics of the learning method the child uses and the complexity of the language, often modeled with a probability distribution over the possible transition from each language L_i to each other (possibly different) L_j. But in the present setting we use the first order approximation that the only incorrect/imperfect learning is a single parameter change per reproductive event. We refer to this constraint as *gradual learning.* The rationale behind this approach is that learning errors do not typically result in the learner acquiring a radically different language. This single parameter change constraint on incorrect/incomplete learning is analogous to only allowing single point mutations to the linear sequence representation of the language. As

such, it defines the "sequence space"[4] through which the population moves during the evolutionary process.

We study language change in an ideal population using a simulation, using the following algorithm. Initially each individual in the population P starts with a randomly chosen language from set of all possible languages.

```
for each individual i ∈ P
    compute fitness f_i of i
end for
do until number of updates is met
    select an individual i ∈ P with a probability proportional to fitness
    select a second random individual j from the population
    replace individual j with an offspring k of individual i
        if the offspring is mutant( mutation rate = μ)
            change a random parameter of L_k
        else
            Lk = L_i
        end if
        update fitness of the individual j
end do
```

We measure the *dominant* language frequency directly at each time step by counting the number of individuals speaking each language. The dominant language at any given time is simply the language that is most frequent at that time, and will typically change over time unless the population has strongly converged.

The linguistic coherence of the population, denoted ϕ, is defined as follows:

$$\phi = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} \tag{2}$$

Counting the actual number of languages that exist in the population may disguise the degree of variation when some of the languages disproportionately dominate. Consequently, we used an analogue to the effective number of alleles in a population, which we will refer to as the effective number of languages in the population, n_e [3]:

$$n_e = \left(\sum_{i=1}^{N} p_i^2 \right)^{-1} \tag{3}$$

where p_i is the frequency of each language.

Table 1 shows the parameter settings for the experimental setup. We used a population size N of 500, a base fitness f_0 of 0.001, and we let the number of different possible languages n be 64. Each language in a set can be represented as a linear sequence of length l with elements drawn from a set of d possible values. For set A, the similarity between languages that are not the same is set to a constant value a equal to 0.5. For all other sets, a_{ij} is the Hamming distance divided by sequence length as described above. The reproduction cycle repeated for 100,000 times to make each run long enough to reach an equilibrium. Twenty

Table 2. System settings, average language similarity ($\bar{a}$), q_1 and q_2. When $l = 1$, we use $a = 0.5$. A: one component with 64 options, B: two components with 8 options, C: three components with 4 options, D: 6 components with 2 options. Each setup has exactly the same number of possible languages.

Setting	l	d	n(= d^l)	$\bar{a}$	q_1	q_2
A	1	64	64	0.500	0.830	0.985
B	2	8	64	0.111	0.516	0.902
C	3	4	64	0.238	0.662	0.955
D	6	2	64	0.492	0.826	0.985

replica runs, varying only the random number generator seed, were done at each q between 0.5 and 1 at 0.02 intervals.

3 Analytic Model

Given a uniform similarity a between n different languages, and the learning fidelity of q, three equilibrium solutions, X_0 and $X_\pm$, for language frequency were derived by Komarova *et. al.*[12] for a family of single-component languages:

$$X_0 = 1/n \tag{4}$$

$$X_\pm = ((a-1)(1+(n-2)q) \mp \sqrt{D})(2(a-1)(n-1))^{-1} \tag{5}$$

where

$$D = 4[1+a(n-2)+f_0(n-1)](1-q)(n-1)(a-1) + (1-a)^2[1+(n-2)q]^2$$

Below a certain learning fidelity of q_1, only the symmetric solution X_0 exists and no single language dominates. Solving for q when $D = 0$ determines the critical leaning fidelity threshold q_1, which corresponds to the *error threshold* in molecular evolution.

$$q_1 = \frac{1}{(1-a)(n-2)^2}\Big[4 + 2(n-1)^{\frac{3}{2}}\sqrt{(1+f_0)\,[1+a(n-2)+f_0(n-1)]} \\ -2f_0(n-1)^2 - 3n - a(2n^2-7n+6)\Big] \tag{6}$$

When $q_1 < q < q_2$ for a specific q_2, both the symmetric $X_\pm$ and asymmetric X_0 solutions exist and are stable. For $q > q_2$ however, only the asymmetric solution where one language dominates the population is stable. This q_2 value is the point where $X_0 = X_-$, giving:

$$q_2 = \left(n^2(f_0+a) + (n+1)(1-a)\right)\left(n^2(f_0+a) + 2n(1-a)\right)^{-1} \tag{7}$$

Komarova *et. al.* provide much more detail and proofs[12].

By introducing a regularity in language, we effectively change the transition matrix of a_{ij}. To compare our findings with the analytical result, we use the average language similarity $\bar{a}$ for calculating q_1 and q_2, where $\bar{a}$ is calculated using the equation below:

$$\bar{a} = \frac{1}{n-1}\left(\sum_{k=1}^{l-1} \frac{l-k}{l}(d-1)^k \binom{n}{k}\right) \tag{8}$$

We consider 4 settings A-D, varying in the "amount of structure." The four cases are listed in Table 2 together with the calculated $\bar{a}$ for each case.

4 Results

We plot the experimental and analytic results for comparison in Figure 1. The empirical results for the uniform similarity of $a = 0.5$ between two different languages closely follows the expectation from the analytic results arrived at by Komarova *et. al.*[12] as shown in Figure 1 A, which we have previously described in detail[14].

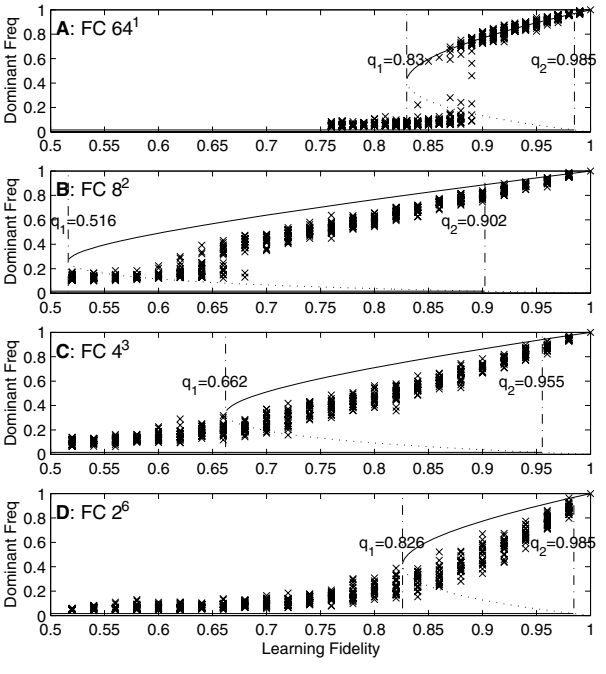


Fig. 1. The dominant(×) language frequency after 100,000 time steps overlaid with symmetric (horizontal line) and asymmetric (curved line) solutions for a(or $\bar{a}$), $n = 64$, $f_0 = 0.001$. Each point is an independent replica. d^l shown at the top left corner of each graph.

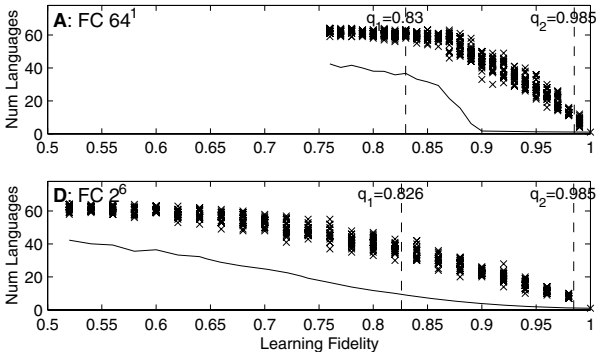


Fig. 2. The number of languages(×) and the average effective number of languages(—)

The results of the multi-component languages (Figure 1 B, C and D) do not show the clear transition from symmetric to asymmetric solution. The trend is considerably smoother, with nothing but an increased variance in results at the point of the phase transition for parameter sets C and D. Parameter set B shows a region where both symmetric and asymmetric solutions appear stable for q values between 0.6 and 0.7, but it is notable that the empirical asymmetric dominant abundance is significantly below the analytical expectation for this set as well as C and D.

Since the setup A and D have similar $\bar{a}$ values ($\bar{a_A} \simeq \bar{a_D}$), they provide a better example of what difference the multi-parameter language brings to the language evolution scenario. Figure 2 compares the number of languages and the effective number of languages (n_e), calculated using the equation (3). In the single-parameter language case A, all the possible languages exist in the population in the region where $q < q_{1A}$. On the other hand, the 6-parameter case D has only half of the all possible languages at $q = q_{1D}$.

Figure 1 A shows that if the learning fidelity is greater than 0.9, one language dominates in the population. That trend is illustrated clearly by the average effective number of languages in Figure 2 A. There are still over half of all possible languages remaining in the population at $q = 0.9$. This number overestimates the true variation in the population when some languages disproportionately dominate while most are at very low frequency. Incomplete/incorrect learning provides a constant influx of variants, but these variants do not propagate to any appreciable frequency due to their inferior fitness. The effective number of languages n_e for the set A at $q = 0.9$ is close to 1 ($n_e = 1.68$), which indicates that the population has converged to one language, and the rest of languages exist a very low frequency .

In contrast, Figure 2 D shows a gradual decline in number of languages as learning fidelity increases. For this set, the number of languages in the population starts decreasing noticeably for q values above 0.55, and the effective number of languages n_e decreases over the entire range. However, at q values above 0.9, set D shows a higher n_e value (3.75 at $q = 0.9$) than set A, indicating that there

are more relatively high abundance languages in set D despite the fact that the total number of languages is lower.

In set A, all possible languages are a single step away in sequence space; in other words, all possible languages are reachable by a single incorrect/incomplete learning event. In set D, however, only a small subset of possible languages are producible as single step variants from the dominant language. These single-step variants of the dominant account for the majority of non-dominant languages in the population. Additionally, these variants have a high fitness relative $\bar{a}$, and a higher equilibrium frequency in mutation-selection balance.

5 Discussion

For the set of single-component languages, our empirical results closely match the analytic results produced by Komarova *et al.* In an unstructured language space, high fidelity learner-driven change, such as the sort exhibited by human languages, can only occur just above the critical error threshold q_1, near the bifurcation point.

These simulations show that substantial levels of linguistic coherence can be achieved with lower learning fidelity if structure is introduced. All four settings explored here have language spaces of exactly the same size, and yet the structured language sets allow fairly stable asymmetric solutions even with quite low learning fidelity and show a much more gradual approach to coherence.

We conclude that a simple regularity combined with gradual learning can dramatically reduce the number of languages that exist in the population, even in regions where analytic results indicate that only symmetric solutions will be stable. Gradual learning used in this experiment seems a more realistic approximation to reality than the "memoryless" learning used in previous work. The qualitatively different dynamics with respect to the critical learning fidelity suggests that convergence to a set of closely related languages is significantly easier than previously thought.

These results are in keeping with the expectations of a quasi-species interpretation. Gradual learning maps the grammars into a sequence space where some grammars have fewer mutational (incomplete/incorrect learning) steps from others. Calculating the similarity between grammars which determines fitness as one minus Hamming distance divided by sequence length ensures that grammars that are close in the sequence space have similar fitness values. This produces a smooth fitness landscape.

The upshot of this smooth fitness landscape is that selection operates on the quasi-species formed by the dominant grammar and its close variants. At learning fidelity values below q_1, the population converges not to a single dominant grammar with all other grammars equally represented, but instead to a family of similar grammars. The fidelity of this family is higher than the nominal learning fidelity because a sizable proportion of incomplete/incorrect learning events among members of the quasi-species result in other members of the quasi-species. At still lower q values, that family of grammars (the quasi-species) spreads far-

ther out in sequence space, until at some point it includes all possible grammars and is identical to the symmetric analytical solution provided by Nowak and Komarova's model [17,12].

For learning fidelity values higher than q_1, we note that a structured grammar space weakens the selection against the minor variants of the dominant grammar in comparison to unstructured or single component grammar models. This effect causes the population to display a dominant abundance below the analytical model's expectations because the close variants of the dominant have a higher equilibrium abundance in mutation-selection balance.

We conjecture natural languages can be viewed as belonging to a highly structured set at some level of description relevant to a theory of learning and cultural transmission, even if this structure is not reducible to a simple sequence representation. As such, the qualitatively different dynamics explored here are important to understanding how human language evolves through time. Additionally, in technological applications where agents learn from each other and it is desirable for the overall system to converge, these results may provide a guide to designing properties of the language or state representation depending on the degree of convergence desired. If it is sufficient that agents of the system just mostly agree, i.e. converge to close variants of a dominant grammar, then a structured state space may provide a way to achieve faster convergence at higher mutation values. However, if absolute convergence is required, the state space must be designed such that minor variants are strongly selected against, producing a sharp fitness peak. This constraint also implies that a critical mutation/learning fidelity threshold exists.

Acknowledgments

This work was supported by NIH, the UCLA Center for Embedded Network Sensors, the Defense Advance Research Projects Agency (DARPA), administered by the Army Research Office under Emergent Surveillance Plexus MURI Award No. DAAD19-01-1-0504, and DARPA MURI award administered by the US Airforce No. F49620-01-1-0361. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

References

1. N. Chomsky. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA, 1965.
2. N. Chomsky. *Rules and Representations*. Basil Blackwell, London, 1980.
3. J. F. Crow and M. Kimura. *An Introduction to Population Genetics Theory*. Harper & Row Publishers, New York, Evanston and london, 1970.
4. M. Eigen, J. McCaskill, and P. Schuster. The molecular quasi-species. *Adv. Chem. Phys.*, 75:149–263, 1989.
5. M. Eigen and P. Schuster. *The hypercycle: A principle of natural self-organization*. Springer Verlag: Berlin, 1979.

6. W. T. Fitch, M. D. Hauser, and N. Chomsky. The evolution of the language faculty: Clarifications and implications. *Cognition*, Forthcoming, 2005.
7. T. Hashimoto and T. Ikegami. *Evaluation of symbolic grammar systems*. Springer-Verlag:Berlin, 1995.
8. T. Hashimoto and T. Ikegami. Emergence of net-grammar in communicating agents. *BioSystems*, 38:1–14, 1996.
9. M. D. Hauser, N. Chomsky, and W. T. Fitch. The faculty of language: what it is, who has it, and how did it evolve? *Science*, 298:1569–1579, 2002.
10. B. Joseph. Historical linguistics. In M. Aronoff and J. Rees-Miller, editors, *The Handbook of Linguistics*. Blackwell, Oxford, 2000.
11. S. Kirby. Spontaneous evolution of linguistic structure: an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation*, 5(2):102–110, 2001.
12. N. L. Komarova, P. Niyogi, and M. A. Nowak. Evolutionary dynamics of grammar acquisition. *Journal of Theoretical Biology*, 209(1):43–59, 2001.
13. S. M. Lamb and E. D. Mitchell. *Sprung from a Common Source: Investigations into the Prehistory of Languages*. Stanford University Press, Stanford, California, 1991.
14. Y. Lee, T. C. Collier, E. P. Stabler, and C. E. Taylor. The role of population structure in language evolution. In *The 10th International Conference on Artificial Life and Robotics*, Beppu, Oita, Japan, 2005.
15. P. Niyogi. *The Computational Nature of Language Learning and Evolution*. MIT Press, Cambridge, Massachusetts, 2003.
`http://people.cs.uchicago.edu/~niyogi/Book.html`.
16. M. A. Nowak, N. Komarova, and P. Niyogi. Computational and evolutionary aspects of language. *Nature*, 417:611–617, 2002.
17. M. A. Nowak, N. L. Komarova, and P. Niyogi. Evolution of universal grammar. *Science*, 291:114–118, 2001.
18. S. Pinker. *The Language Instinct*. Penguin, London, 1994.
19. S. Pinker and P. Bloom. Natural language and natural selection. *Behavioral and Brain Sciences*, 13:707–784, 1990.
20. K. Smith, H. Brighton, and S. Kirby. Complex systems in language evolution: the cultural emergence of compositional structure. *Advances in Complex Systems*, 6(4):537–558, 12 2003.
21. Staff. Endangered languages: Babel runs backwards. *Economist*, 374(8407), January 2005.
22. L. Steels. Self-organizing vocabularies. In C. Langton and T. Shimohara, editors, *Artificial Life V*, pages 179–184, Nara, Japan, 1996.
23. W. J. Sutherland. Parallel extinction risk and global distribution of languages and species. *Nature*, 423:276–279, 2003.

Interactions Between Learning and Evolution in Simulated Avian Communication

Edgar E. Vallejo[1] and Charles E. Taylor[2]

[1] ITESM-CEM, Computer Science Dept.,
Atizapán de Zaragoza, Edo. de Méx., 52926, México
vallejo@itesm.mx
[2] UCLA, Dept. of Ecology and Evolutionary Biology,
Los Angeles, CA, 90095-1606, USA
taylor@biology.ucla.edu

Abstract. This paper presents a computational framework for studying the influence of learning on the evolution of avian communication. We conducted computer simulations for exploring the effects of different learning strategies on the evolution of bird song. Experimental results show the genetic assimilation of song repertoires as a consequence of interactions between learning and evolution.

1 Introduction

The evolution of avian communication is an excellent domain for studying fundamental questions of artificial life research. Previous work by Sasahara and Ikegami [14][15], have shown that we are able to explore important issues such as emergence, self-organization and cultural evolution within this framework. Similarly, artificial life models provide a convenient alternative to complex playback and genetic experiments for validating theories of bird song evolution by means of computer simulations.

Bird song studies have been established as instrumental in resolving the debate over instinct versus learning in the ontogeny of behavior [3]. There is a wide variety of patterns in the development of song. For example, among the suboscines normal song develops in individuals that are isolated and or even deafened at an early age. In contrast, among oscines, individuals typically need an external model and intact hearing for normal song development to occur[10]. In addition, birds have been excellent subjects for studying how signals are transmitted and perceived in noisy environments and how the structure of vocalizations can be optimized to achieve these goals [9]. We believe these studies are crucial for understanding the origin and evolution of communication systems with the complexity of human languages.

The aim of this work is to study the effects of learning on the evolution of avian communication using computer simulations. To this end, we formulate a computational framework based on the seminal model proposed by Hinton and Nowlan [8] and further developed by Ackley and Littman [1], among others. In addition, we explore the effects of a noisy communication channel on the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 634–643, 2005.

evolution of bird song within the proposed framework. Experimental results show that communicative behaviors become innate as a consequence of interactions between learning and evolution.

2 The Model

2.1 Environment

In our model, the environment consists of a population of communicative agents $\mathcal{A}$. This population representes a simulated bird species. The environment may include other simulated bird species $\mathcal{B}_i$ that sing different songs with respect to $\mathcal{A}$.

2.2 Agent Architecture

In our model, a simulated bird consists of an agent arquitecture that represents his song repertoire. The formal definition of the agent architecture presented below is based on considerations of the model proposed by Vallejo and Taylor [16].

Agent. Let $S = \{s_1, \ldots, s_n\}$ be a finite set of n songs and $R = \{r_1, \ldots, r_m\}$ be a finite set of m external referents. An *agent* A is a pair (δ, ϕ), where

1. $\delta : R \rightarrow S \cup \{s_\#\}$ is the transmission function, where $s_\#$ is the undetermined song, and
2. $\phi : S \rightarrow R \cup \{r_\#\}$ is the reception function, where $r_\#$ is the undetermined referent.

Communication. An agent $A_1 = (\delta_1, \phi_1)$ comunicates to an agent $A_2 = (\delta_2, \phi_2)$ as follows. Initially, A_1 perceives the referent r_i and produces a song s_j according to the mapping described by the transmission function δ_1, such that $\delta_1(r_i) = s_j$. Once A_1 produces the song s_j, the agent A_2 interprets the song s_j as the referent r_k according to the mapping described by the reception function ϕ_2, such that $\phi_2(s_j) = r_k$. A communication event from A_1 to A_2 is successful if the following conditions are satisfied:

1. $\delta_1(r_i) = s_j$,
2. $\phi_2(s_j) = r_k$, and
3. $r_i = r_k$

Innate Transmission. Let $A = (\delta, \phi)$ be an agent. A transmission from A for a given referent r_i is said to be innate if $\delta(r_i) \neq s_\#$ and is said to be subject to learning if $\delta(r_i) = s_\#$.

Innate Reception. Let $A = (\delta, \phi)$ be an agent. A reception of A for a given song s_j is said to be innate if $\phi(s_j) \neq r_\#$ and is said to be subject to learning if $\phi(s_j) = r_\#$.

Learning. In our model, both transmission and reception behaviors of an agent are partially learned. Before a communication event from A_1 to A_2 takes place, A_1 replaces the undetermined songs in δ_1 with songs in S using a predefined learning strategy. Similarly, A_1 replaces the undetermined referents in ϕ_1 with referents in R using the learning strategy. A_2 proceeds similarly.

A fundamental aspect of our model is that learning is performed for communication purposes and does not modify permanently the actual description of an agent. In other words, learned characteristics are not transmited to offspring during reproduction.

2.3 Learning Strategies

We consider two different learning strategies: imitator and improviser, as they are two main forms of bird song learning [7][11]. These strategies are described below.

Imitator. An imitator learner replaces the undetermined songs in his trasmission function by the corresponding songs in the transmission function of a teacher. Similarly, he replaces the undetermined referents in his reception function by the corresponding referents in the reception function of a teacher.

Formally, a learner $A_1 = (\delta_1, \phi_1)$ imitates a teacher $A_2 = (\delta_2, \phi_2)$ as follows.

1. $\delta_1(r_i)$ is set to $\delta_2(r_i)$ if $\delta_1(r_i) = s_\#$, $\delta_2(r_i) \neq s_\#$, for $i = 1, \ldots, n$, and
2. $\phi_1(s_j)$ is set to $\phi_2(s_j)$ if $\phi_1(s_j) = r_\#$, $\phi_2(s_j) \neq r_\#$, for $j = 1, \ldots, m$.

Improviser. An improviser learner replaces the undetermined songs in his transmission function by random songs in S. Similarly, he replaces the undetermined referents in his reception function by random referents in R.

Formally, a learner $A_1 = (\delta_1, \phi_1)$ improvises as follows.

1. $\delta_1(r_i)$ is set to $random(S)$ if $\delta_1(r_i) = s_\#$, for $i = 1, \ldots, n$, and
2. $\phi_1(s_j)$ is set to $random(R)$ if $\phi_1(s_j) = r_\#$, for $j = 1, \ldots, m$.

2.4 Evolution of Communication

In our model, a population of simulated birds are intended to evolve successful communication at the population level. We use genetic algorithms for this purpose. The design decisions presented below are based on considerations of the performance of genetic algorithms in practical applications [13].

Genome Representation. An agent $A = (\delta, \phi)$ is represented linearly as follows

$$A = (\delta(r_1), \ldots, \delta(r_n), \phi(s_1), \ldots, \phi(s_m))$$

Genetic Operators. Agents produce a new offspring by means of genetic operators. Fitness proportional selection, one-point recombination and point mutation operate on the linear representation of agents described above.

Fitness Function. Fitness is defined as the communicative accuracy of agents. The communicative accuracy is the ability of an agent to successfully communicate with a collection of other agents.

Let P be a finite population of agents, A be an agent in P, and $Q \subseteq P$ be a non empty collection of agents. The communicative accuracy of A with respect to Q given the set of referents $R = \{r_1, \ldots, r_n\}$ and the set of songs $S = \{s_1, \ldots, s_m\}$, $C(A, Q, R)$, is defined as

$$C(A, Q, R) = \frac{\sum_{r_i \in R} \sum_{A_k \in Q} c(A, A_k, r_i) + c(A_k, A, r_i)}{|Q|}$$

where $c(A, A_k, r_i) = 1$ if the communication event from A to A_k is successful given the referent r_i, and 0 otherwise; $|Q|$ is the cardinality of Q. $c(A_k, A, r_i)$ is defined similarly.

There is evidence of both temporal song avoidance and song divergence in neighbouring bird species [5][6]. We consider this fact in our model as follows.

If there exist other simulated bird species $\mathcal{B}_i$ in the environment, then a distance component is added to the fitness value defined above. The distance between an agent A and other simulated bird species $\mathcal{B}$ is defined as follows

$$D(A, \mathcal{B}) = \frac{\sum_{\mathcal{B}_i \in \mathcal{B}} H(A, \mathcal{B}_i)}{|\mathcal{B}|}$$

where $H(A, \mathcal{B}_i)$ is the Hamming distance between A and $\mathcal{B}_i$.

Therefore, the fitness of an agent A is defined as

$$f(A) = C(A, Q, R) + D(A, \mathcal{B})$$

3 Experiments and Results

A series of experiments were conducted to investigate whether a population of simulated birds is likely to arrive to successful communication at the population level. In addition, we validated the evolutionary performance of competing learning strategies. Most importantly, we were interested in exploring the effects of learning on the genetic description of an evolving population of simulated birds. Finally, we explore the influence of different levels of noise in the communication channel on the genetic assimilation of traits. The simulation procedure is described in table 1.

Extensive simulations were conducted using different combinations of parameter values as shown in table 2. The following were the major results:

1. In one-strategy simulations, imitators arrived to highly accurate communication at the population level. On the other hand, improvisers reached local maxima in communication accuracy consistently. Figure 1 shows the results of representative simulations of the two learning strategies.
2. In one-strategy simulations, imitation produced the genetic assimilation of both songs and referents. On the other hand, improvisation reduced the undetermined songs and referents but they were not totally assimilated. Figure 2 and figure 3 show the frequency of undetermined songs and undetermined referents in the population of the two learning strategies, respectively.

Table 1. Simulation procedure

1. Create an initial random population P of agents
2. `Do until` a predefined number generations is met
   (a) `For each` individual $A_i = (\delta_i, \phi_i) \in P$ `do`
      i. Perform the learning process of A_i according to the learning strategy with respect to a random agent $A_h \in P$
      ii. Select a random subpopulation of agents $Q \subseteq P$
      iii. Perform the learning process for all $A_j \in Q$ according to the learning strategy of A_j with respect to a random agent $A_k \in P$
      iv. Measure the communicative accuracy of A_i with respect to Q, $C(A_i, Q, R)$, given the set of referents R
      v. Measure the distance of A_i with respect to $\mathcal{B}$, $D(A_i, \mathcal{B})$, for all extant species $\mathcal{B}_i$
      vi. Compute the fitness $f(A_i) = C(A_i, Q, R) + D(A_i, \mathcal{B})$
   `End for`
   (b) Select two individuals $A_{mother} \in P$ and $A_{father} \in P$ for reproduction using fitness proportional selection
   (c) Produce an offspring A_{new} from A_{mother} and A_{father} using one-point recombination and point mutation
   (d) Select a random individual $A_{old} \in P$
   (e) Replace A_{old} by A_{new}
`End do`

Table 2. Parameters for simulations

Parameter	**Value**
undetermined traits	50%
generations	3000
population P	256
subpopulation Q	16
songs S	4-8
referents R	4-8
crossover probability P_c	0.6
mutation probability P_m	0.001
species $\mathcal{B}$	0-32

3. In two-strategy simulations, a population of imitators dominated a population of improvisers. In most cases, imitators took over the entire population. Very rarely, a few improvisers prevailed in the population. Figure 4 shows the frequency of strategies in the population of a typical two-strategy simulation.
4. In one-strategy simulations when there were other species present in the environment, a densely populated environment produced a faster genetic assimilation of both songs and referentes. Figures 5 and 6 show the frequency of undetermined songs and undetermined referents in the population of a characteristic two-strategy simulation with different number of species present in the environment, respectively.

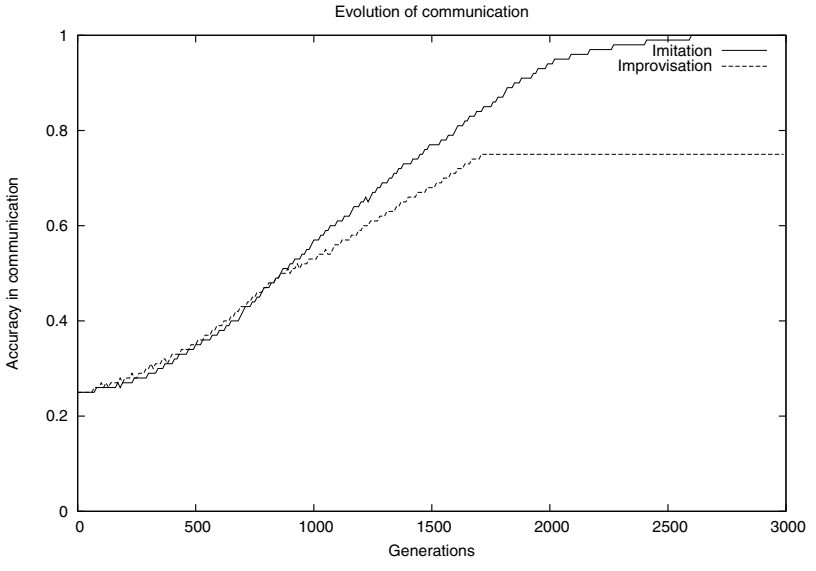


Fig. 1. Evolution of communication in one-strategy simulations

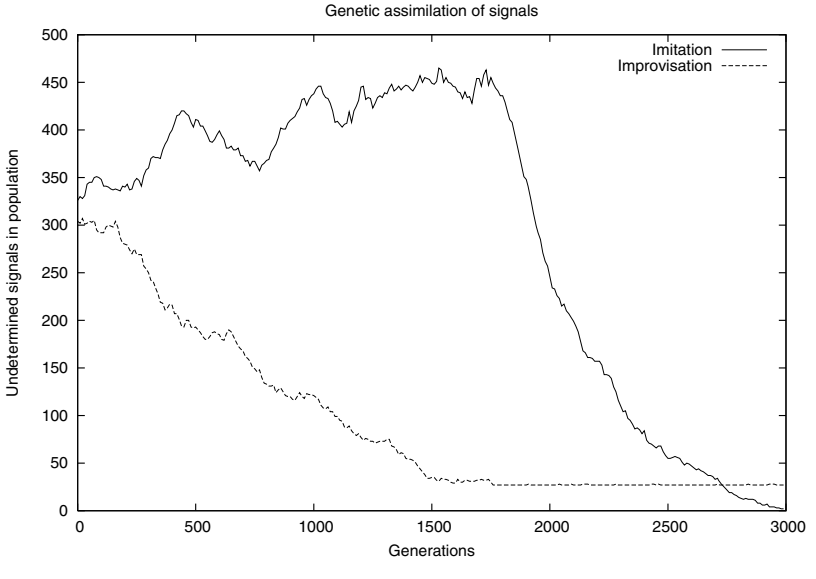


Fig. 2. Genetic assimilation of songs in one-strategy simulations

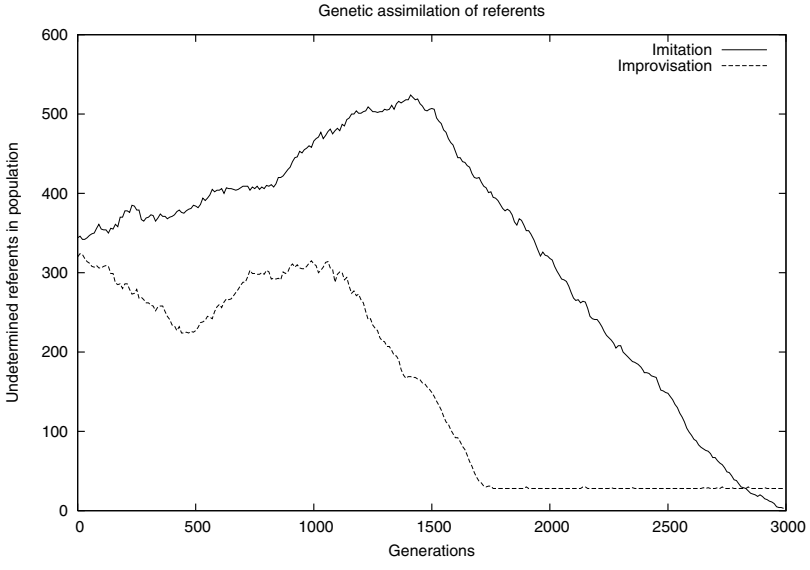


Fig. 3. Genetic assimilation of referents in one-strategy simulations

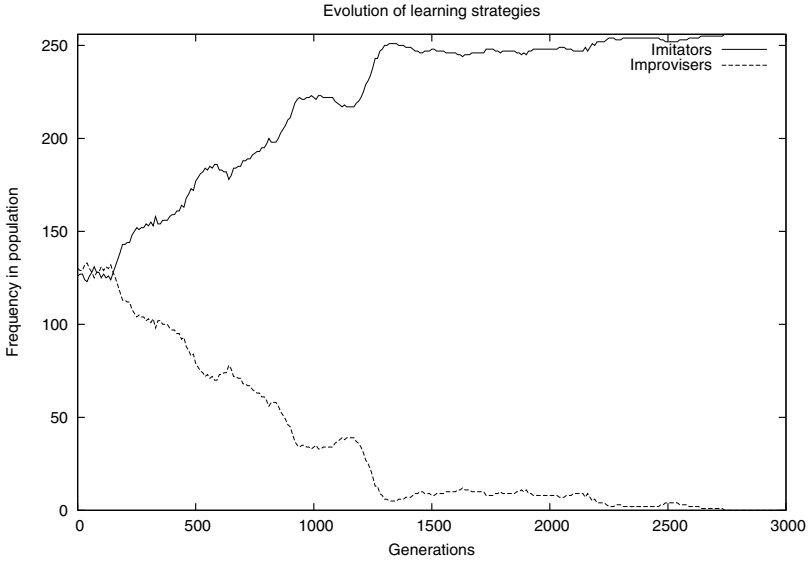


Fig. 4. Frequency of learning strategies in two-strategy simulations

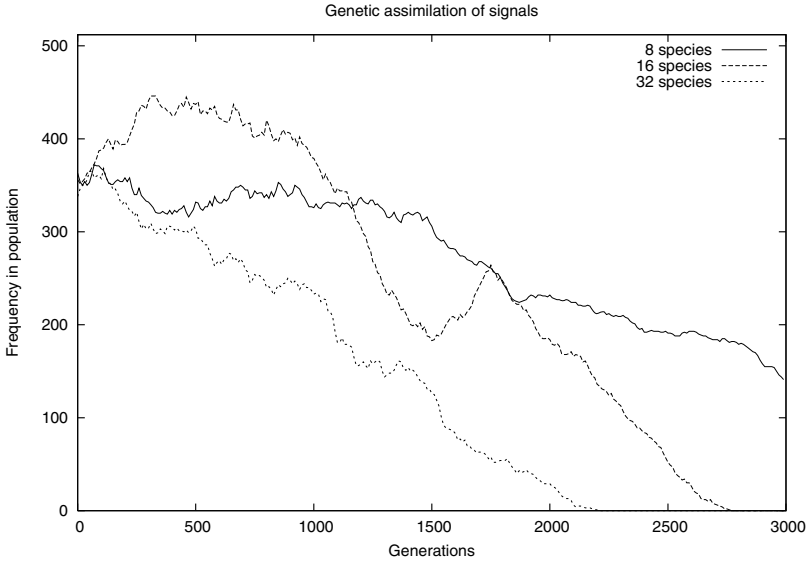


Fig. 5. Genetic assimilation of songs in noisy environment

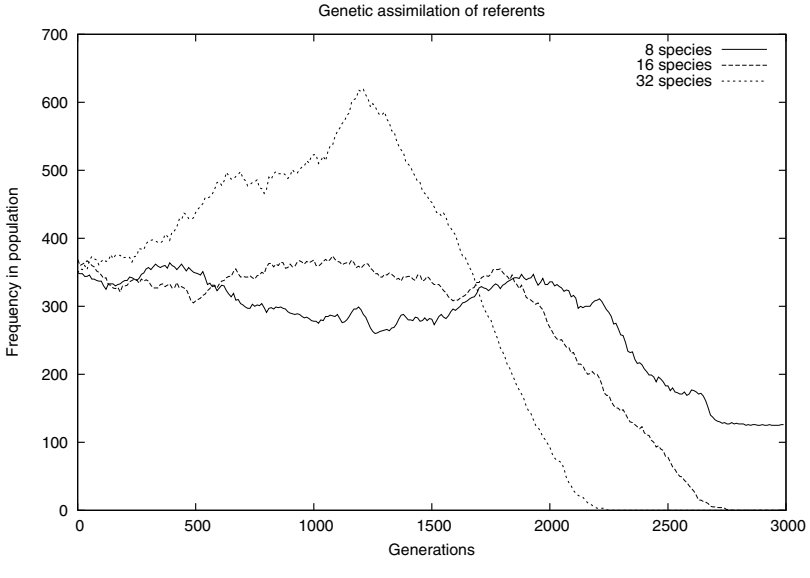


Fig. 6. Genetic assimilation of referents in noisy environment

4 Discussion

Overall, experimental results indicate that a population of agents is capable of arriving to highly successful communication. Both transmission and reception behaviors became innate as a consequence of the interaction between learning and evolution.

Why communicative behaviors became innate? First, imitation of conspecifics in a static environment provides the opportunity for the genetic assimilation of transmission and reception behaviors. Second, the competition for the communication channel contributes to accelerate the assimilation of traits. There are examples of similar innate underpinnings in bird song [12].

So far, we have not considered the cost of producing a song. The fundamental issue of honesty would arise as a result of this consideration. Previous artificial life studies have provide insights on this topic [2].

We believe that the proposed framework could also be used for testing theories on allopatric speciation, song sharing, stability of song types [4]. These studies would contribute to elucidate the origins and evolution of bird song.

Acknowledgements

This work was supported by US National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

References

1. Ackley, D.H. and Littman, M.L.: Interactions between learning and evolution. In C.G. Langton, C.E. Taylor, J.D. Farmer and S. Rasmussen *Artificial Life II.* (1992) Addison Wesley.
2. Ackley D,H. and Littman, M.L.: Altruism in the Evolution of Communication. In R.A. Brooks, P. Maes, (eds), Artificial Life IV. Proceedings of the Fourth International Conference on Artificial Life. (1994) The MIT Press.
3. Baptista, L.F.: Nature and nurturing in avian vocal development. In [10].
4. Catchpole, C. K., Slater, P.J.B.: Birsong Biological Themes and Variations. (2003) Cambridge University Press.
5. Cody, M.L.: Song asynchrony in neighbouring bird species. Nature 222(1969): 778–780.
6. Gill, F.B., Murray, B.G.: Song variation in sympatric blue-winged and golden-winged warblers. The Auk 89:625–643 (1972).
7. Green, E.: Toward an evolutionary understanding of song diversity in oscines. The Auk 116 (2):299-301 (1999).
8. Hinton, G.E. and Nowlan, S.J.: How learning can guide evolution. Complex Systems**1** (1992): 495–502.
9. Klump, G.M: Bird communication in the noisy world. In [10].
10. Kroodsma, D.E., Miller, E.H. (eds.): Ecology and Evolution of Acoustic Communication in Birds. (1996) Comstock Publishing.

11. Kroodsma, D.E., Liu W.C., Goodwin, E., Bedell, P.A.: The ecology of song improvisation as illustrated by north american sedge wrens. The Auk 116(2):373–386 (1999).
12. Kroodsma, D.E: The song of the alder flycatcher and willow flycatcher are innate. The Auk 101:13-24 (1984).
13. Mitchell, M. : An introduction to genetic algorithms. (1996) The MIT Press.
14. Sasahara, K. and Ikegami, T.: Song Grammars as Complex Sexual Displays. In J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R.A. Watson (eds.) Artificial Life IX. Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems. (2004) The MIT Press.
15. Sasahara, K. and Ikegami, T.: Coevolution of Birdsong Grammar without Imitation In W. Banzhaf, T. Christaller, P. Dittrich, J.T. Kim, J. Ziegler (Eds.), ECAL 2003, Advances in Artificial Life. 7th European Conference (2003) LNAI Vol. 2801.
16. Vallejo, E.E., Taylor, C.E.: The effects of learning on the evolution of Saussurrean communication. In J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R.A. Watson (eds.) Artificial Life IX. Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems. (2004) The MIT Press.

Perceptually Grounded Lexicon Formation Using Inconsistent Knowledge

Federico Divina[1] and Paul Vogt[1,2]

[1] Computational Linguistics and AI Section,
Tilburg University, Tilburg, The Netherlands
F.Divina@uvt.nl
[2] Language Evolution and Computation, (LEC) Unit,
University of Edinburgh - UK
paulv@ling.ed.ac.uk

Abstract. Typically, multi-agent models for studying the evolution of perceptually grounded lexicons assume that agents perceive the same set of objects, and that there is either joint attention, corrective feedback or cross-situational learning. In this paper we address these two assumptions, by introducing a new multi-agent model for the evolution of perceptually grounded lexicons, where agents do not perceive the same set of objects, and where agents receive a cue to focus their attention to objects, thus simulating a Theory of Mind. In addition, we vary the amount of corrective feedback provided to guide learning word-meanings. Results of simulations show that the proposed model is quite robust to the strength of these cues and the amount of feedback received.

1 Introduction

In the past decade, a number of studies have investigated the emerge of perceptually grounded lexicons in multi-robot systems [14,13]. The aim of such studies is to investigate under what conditions a population of (possibly simulated) robots can evolve a shared vocabulary (or *lexicon*) that allow them to communicate about objects that are in their environment. Typically, these studies assume that the lexicons evolve culturally through inter-agent communication, individual adaptation and self-organisation [12]. These *perceptually grounded* studies extend other *ungrounded* models in which the meanings of the lexicons are predefined as in, e.g., [6,8], by allowing the agents to develop their meanings from scratch based on their sensing of the environment. Typically, the lexicons and meanings are constructed through *language games* [12] in which two agents communicate about an object they detect in a certain context. This way, the grounded models add more realism to the ungrounded models, as they do not assume that agents have an innate set of meanings.

The grounded models, however, still build upon many assumptions. One such assumption is that agents perceive the same set of objects. Especially in simulations – both grounded [10,17] and ungrounded [8,5,18] – this is taken for granted. In studies using real robots, this is assumed, though not necessarily achieved [16]. When two agents communicate, one cannot simply assume that

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 644–654, 2005.

the context of one agent is similar to the other agent's context. Agents located in different places see different things – even if they are located close to each other and looking in a similar direction.

The problem agents face when learning the meaning of words is that when hearing a novel word, logically, this word can have an infinite number of meanings or references – even if the target is pointed at [9]. So, in order to learn the meaning of a word, an agent has to reduce the number of possible meanings. Humans are exceptionally good at this, and it is generally assumed that humans have some innate or acquired means to infer what a speaker's intention is, see, e.g., [2] for an overview. Among these means are *joint attention* [15], *Theory of Mind* (ToM) [2], and receiving *corrective feedback* on the meaning of words [3]. In joint attention, the participants of a communication act focus their attention to an object or event while actively checking that they share their attention. In a way, this requires that the participants understand each other as having similar intentions [15]. Loosely speaking, this is a part of the Theory of Mind. However, ToM is more: it allows someone to form theories regarding the intentions of speakers by simulating that he or she is the speaker (him)herself. For instance, a child may know that its caregiver is hungry and can therefore infer the caregiver is more interested in food than in a doll. The problem with joint attention and ToM is that it is not always precise. Suppose a rabbit passes by and someone shouts 'gavagai', even if you establish joint attention, you cannot be sure 'gavagai' refers to the rabbit; it may also refer to undetached rabbit parts or that it is going to rain [9]. To further reduce the number of possible meanings for a word, caregivers sometimes provide corrective feedback on the meaning of childrens' utterances. Although the availability of corrective feedback is highly disputed [2], recent analysis has shown that it is actually abundant, especially with respect to the meaning of words [3]. However, corrective feedback is not always present. In those cases, we assume that children can learn the meaning of words across situations by focusing on the covariance between word meaning pairs. There is some recent evidence that children indeed use such a learning strategy [1].

The current study addresses the two issues discussed and introduces a model in which a perceptually grounded lexicon is developed by a population of simulated robots based on the Talking Heads experiment [13]. In this model, the contexts that agents perceive while playing a language game differ from each other. In addition, as suggested in [16,18], the three models are integrated together with a naive implementation of ToM. The ToM is implemented by introducing attention cues that focus the (possibly joint) attention on objects in a context. In this study, these cues are assigned randomly, but in future work we intend to implement this by having the agents estimate these cues autonomously based on some knowledge about intentions [19]. Using these cues and a verbal hint provided by the speaker, the hearer will guess the reference of the uttered word, and in some cases the agents evaluate corrective feedback. In addition, co-occurrence frequencies are maintained to allow for cross-situational statistical learning [20]. The experiments reported in this paper investigate the effect

of using these attention cues and to what extent the model is still robust when we vary the amount of corrective feedback available to the learners.

The next section introduces the new model. In Section 3 we experimentally assess the validity of the proposed model, and finally Section 4 concludes.

2 The Model

The model we propose here is implemented in the Talking Heads simulation toolkit THSim[1] [17] and extends the iterated learning model (ILM) [7] in combination with the language game model [13]. The ILM implements a generational turnover in which a population of adults transmits its acquired lexicon to the next generation of learners culturally by engaging in a series of language games. A language game is played by two agents, which are randomly selected from the population at the start of each game: a speaker taken from the adult population and a hearer taken from the learner population. In each generation, a fixed number of language games are played and after each generation, the adults are removed, the learners become adults and new agents enter the population.

Each time a language game is played, both agents a individually observe a context C_a that contains a number of objects o_i. The objects are geometrical coloured shapes, such as red triangles and blue squares. The contexts of the agents share a number of objects, while the rest are distinct. If the contexts contain n objects, the agents share $1 \leq k \leq n$ objects, where k is assigned randomly in each game. If $k = n$, then the contexts are equal.

In order to provide each agent a with some naive form of Theory of Mind, we simulate the use of *attention cues* $strA(o_i)$ assigned to each object $o_i \in C_a$. In future models [19], we intend to base these attention cues on a more sophisticated ToM, which the speaker uses to select the topic of the language game and which the hearer uses to estimate the speaker's intention (see Section 2.2). For the moment we assume that these attention cues are assigned with random values, where shared objects have a high attention cue, while objects that are not shared possess a low attention cue. In short:

$$strA(o_i) = X_i = \begin{cases} \beta_s \leq X_i \leq 1 & \text{if } o_i \text{ is shared,} \\ 0 \leq X_i \leq \beta_u & \text{if } o_i \text{ is not shared.} \end{cases} \quad (1)$$

where β_s and β_u are user supplied parameters, with default values of $\beta_s = 0.5$ and $\beta_u = 0.1$. These values were experimentally determined to yield good results; in general it was found that the results were good when $\beta_s > \beta_u$. Shared objects are assigned the same attention cue value.

2.1 Categorising Objects

Categorisation of objects is based on the *discrimination game* model [11] and implemented using a form of 1-nearest neighbourhood classification [4]. The aim

[1] THSim is available from http://www.ling.ed.ac.uk/~paulv/thsim.html

of the discrimination game is to categorise an object (called the *topic* o_t) in an agent's context such that it is distinctive from the other objects in the context. Note that this does not require that the category is a perfect exemplar of the object. By playing a number of discrimination games, each agent a constructs its own ontology O_a, which consists of a set of categories: $O_a = \{c_0, \ldots, c_p\}$. The categories c_i are represented as prototypes $\mathbf{c}_i$ which are points in a n-dimensional conceptual space. Each agent starts its life with an empty ontology.

Each object is perceived by six perceptual features f_q: colour (expressed by Red, Green and Blue components of the RGB space), shape (S) and location (expressed by X and Y coordinates). Each agent a extracts for each object $o_i \in C_a$ a feature vector $\mathbf{f}_i = (f_1 \ldots, f_n)$, where n is equal to the number of perceptual features.

Each object $o_i \in C_a$ is categorised by searching a category $c_j \in O_a$, such that the Euclidean distance $||\mathbf{f}_i - \mathbf{c}_j||$ is smallest. It is then verified that the category found for the topic o_t is distinctive from the categories of the other objects $o_k \in C_a \backslash \{o_t\}$. If no such category exists, the discrimination game fails, and the ontology of the agent is expanded with a new category for which the feature vector $\mathbf{f}_t$ of the topic is used as an exemplar. Otherwise the discrimination game succeeds and the found category is forwarded as the topic's *meaning* m to the production or the interpretation phase.

2.2 The Language Game

The language game we propose, outlined in Table 1, combines the guessing game, e.g., [13] with a *cross-situational statistical learner* [10,16,20]. In both models, the hearer h guesses the topic based on the utterance produced by the speaker, where the topic $o_t \in C_h$. In the guessing game, the agents evaluate whether or not the hearer guessed the right topic (i.e. the object referred to by the speaker). In cross-situational statistical learning (CSSL) such feedback is not evaluated, instead the agent keeps track of co-occurring word-meaning pairs. In order to provide a naive ToM, the model is adapted to include the attentional cues $strA(o_i)$.

In a nutshell, the agents start by perceiving the context of the game and categorise the objects they see using the discrimination game (DG) as explained

Table 1. The outline of a language game, see the text for details

speaker	hearer
-perceive context -categorisation/DG -produce utterance -update $memory_1$ -send message	
	-receive message -perceive context -categorisation/DG -interpret utterance -update $memory_1$
-corrective feedback	-corrective feedback
-update $memory_2$	-update $memory_2$

	m_1	...	m_N
w_1	P_{11}	...	P_{1N}
⋮	⋮	⋮	⋮
w_M	P_{M1}	...	P_{MN}

memory$_1$

	m_1	...	m_N
w_1	σ_{11}	...	σ_{1N}
⋮	⋮	⋮	⋮
w_M	σ_{M1}	...	σ_{MN}

memory$_2$

Fig. 1. Two associative memories constructed and maintained as part of an agent's lexicon. The left memory ($memory_1$) associates meaning m_j with word w_i using conditional a posteriori probabilities P_{ij}. The right memory ($memory_2$) associates meanings m_j with words w_i using an association score σ_{ij}.

above. Then the speaker s selects an object from its context as the topic $o_t \in C_s$ of the language game. In order to so, a roulette wheel mechanism is used, where the sectors of the roulette wheel are proportional to the attention cues $strA(o_i)$ assigned to the objects. Thus, typically, objects that are shared with the context of the hearer have more probability of being selected, since generally their attention cues are higher. As the hearer uses these attentional cues as a bias in guessing the speaker's topic, it is virtually simulating the speaker's selection process. This, we believe, is a naive form of ToM, which in future work we intend to work out more realistically, based on agents' more sophisticated selection criteria.

Each agent maintains an internal lexicon, represented by two associative memories, as illustrated in Figure 1. One of the associative memories (referred to as $memory_1$ in Figure 1) keeps an *a posteriori probability* P_{ij}, which is based on the occurrence frequencies of associations. The other matrix ($memory_2$) maintains an *association score* σ_{ij}, which indicates the effectiveness of an association based on past experiences. The reason for this twofold maintenance is that studies have revealed that when strong attentional cues (such as the corrective feedback used in the guessing game) guide learning, lexicon acquisition is much faster with the association score σ_{ij} than with the a posteriori probabilities [18]. The reverse is true when such strong attentional cues are absent as in CSSL. This is mainly because the update mechanism reinforces the score σ_{ij} more strongly than the update of usage based probabilities P_{ij}. This works well when the cues are precise, but the fluctuations of σ_{ij} would be too strong to allow statistical learning in CSSL.

The probabilities are conditional probabilities, i.e.,

$$P_{ij} = P(m_j|w_i) = \frac{u_{ij}}{\sum_j u_{ij}} \tag{2}$$

where u_{ij} is the co-occurrence frequency of meaning m_j and word w_i. This usage frequency is incremented each time word w_i co-occurs with meaning m_j that is either the topic's meaning (in case of the speaker) or the meaning of an object in the context (in case of the hearer). The update is referred to in Table 1 as 'update $memory_1$'. If this principle is the only mechanism, the learning is

achieved according to the CSSL principle, i.e., across different situations based on the covariance in word-meaning pairs [20].

When corrective feedback is evaluated, the association score σ_{ij} is updated according to the following formula:

$$\sigma_{ij} = \eta\sigma_{ij} + (1 - \eta)X \tag{3}$$

where η is a learning parameter (typically $\eta = 0.9$), $X = 1$ if the association is used successfully in the language game, and $X = 0$ if the association is used wrongly, or – in case of a successful language game – if the association is competing with the used association (i.e., same word, different meaning; or same meaning, different word). The latter implements lateral inhibition. If Eq. (3) is the only update, the game reduces to the guessing game. The update of association scores is referred to in Table 1 as 'update $memory_2$' and is only carried out if corrective feedback is evaluated. The rate with which corrective feedback is evaluated is subject of the second experiment.

Given these two matrices, the speaker, when trying to produce an utterance, calculates an association strength $strL(\alpha_{it})$ for each association α_{it} of a word w_i with the topic's meaning m_t. This is done using Eq. (4):

$$strL(\alpha_{it}) = \sigma_{it} + (1 - \sigma_{it})P_{it} \tag{4}$$

This formula neatly couples the two variables. When σ_{it} is high, the influence of P_{it} is low, and when σ_{it} is low, P_{it} will have more influence. This implements a bias toward basing a choice on known effectiveness vs. estimated probabilities. In Eq. (4), σ_{it} and P_{it} might be weighted, in order to rely more on the association scores or on the a posteriori probabilities. The speaker will select the association that has the highest strength $strL(\alpha_{it})$ and utters its word. If no association can be found, e.g., because the lexicon is still empty, the speaker invents a new word and adds the association to its lexicon with an initial association score $\alpha_{it} = 0.01$ and $u_{it} = 0$.

When the hearer h receives an utterance, it looks in its memories for associations with the current signal and whose meanings match the meanings for each object $o_j \in C_h$ in its context. Using the association strengths $strL(\alpha_{ij})$ and attentional cues $strA(o_j)$, the hearer then interprets the utterance using the following equation based on [5]:

$$\rho_{ij} = \omega_L \cdot strL(\alpha_{ij}) + \omega_A \cdot strA(o_j) \tag{5}$$

where ω_L and ω_A are weights between 0 and 1. Throughout both experiments $\omega_L = 1$ is kept constant, the value of ω_A is subject of variation in the first experiment. If the heard word is not in its lexicon, then the hearer will add it to the lexicon in association with all meanings of the objects in the context and $u_{ij} = 1$. If the agents evaluate the feedback, the word is additionally associated with the meaning m_t of the now-known topic o_t with an initial association score $\sigma_{it} = 0.01$. Feedback is provided to the agents with a given probability P_{fbk}, which is subject to variation in the second experiment. When feedback is provided, the agents update $memory_2$using Eq. (3). The hearer will not update $memory_2$ in the case where the topic is not in its context, since in this case it cannot perceive the category of the topic.

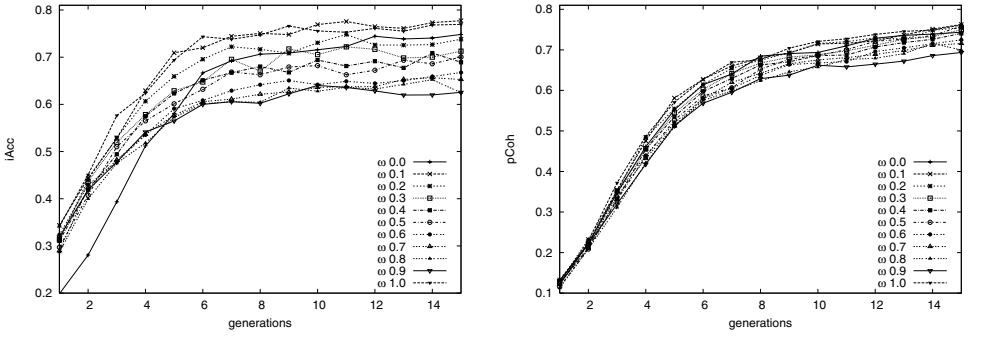


Fig. 2. Results obtained with different values of ω_A. The left figure shows interpretation accuracy, the right one production coherence.

3 Experiments

In order to assess the validity of the proposed model we performed several experiments. In the following we present results using two measures: *production coherence* and *interpretation accuracy*. Production coherence is defined as the fraction of agents that produced the same utterance to name objects. Interpretation accuracy is the fraction of agents that could successfully interpret the produced utterances, averaged over the number of games played. These measures were calculated during a testing phase, consisting of 200 games in which the language did not evolved. The test phase took place at the end of each generation. The results presented are averages over ten runs using different random seeds. In the experiments we used a population of 10 agents in 15 generations of 10,000 language games each. During all experiments the context size $n = 4$ was kept constant, while $1 \leq k \leq n$ was chosen randomly each language game.

A first set of experiments was aimed at evaluating the effect of considering $strA(o_i)$ in Eq. (5) by varying the weight ω_A between 0 and 1 with intermediate steps of 0.1. In this experiment, corrective feedback was always evaluated, i.e., $P_{fbk} = 1.0$. The results for interpretation accuracy are shown in Fig. 2 (left). In this figure the x-axis represents the number of generations processed. It is interesting to notice that the model also yields good results when the attention cues are not considered, i.e. when $\omega_A = 0$ accuracy increases to one of the highest levels. (Note that this setting reduces the model to the guessing game.) In general, when ω_A increases, the results get poorer – especially after about 5 generations So, it seems that agents get confused when higher values of ω_A are used, i.e., the attentional cues suggest a different interpretation than the lexicon. However when $\omega_A = 1$, accuracy is good again, which suggests that the attentional cues can have a positive impact on the learning process, provided they are sufficiently strong. Coherence (Fig. 2 right) reveals a similar evolution, though the values show less variation. At the end of the simulations, coherence is between 0.70 and 0.75. We have further investigated this aspect with equal contexts, i.e., where in all cases $k = n$. For reasons of space, we cannot report

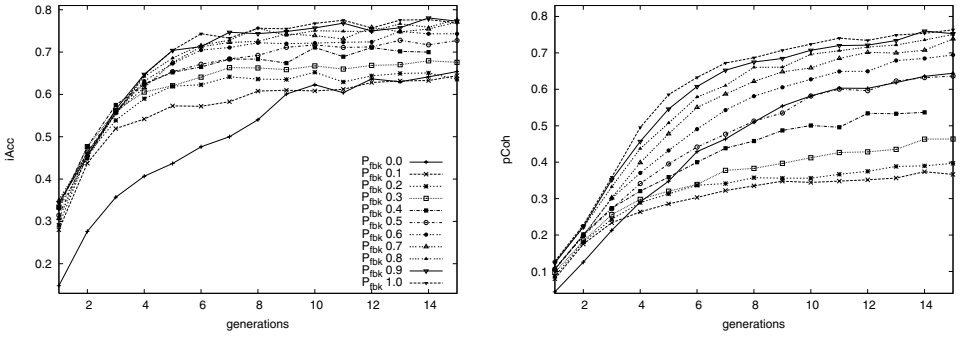


Fig. 3. Results obtained with different values of P_{fbk}. The graphs show interpretation accuracy (left) and production coherence (right).

the results here, however we can say that the results reflect the results presented here. Another set of experiments was performed in order to assess the behaviour of the model when ω_L was varied. The model obtained similar results for all values of this parameter, except when $\omega_L = 0$ the results were considerably worse.

Another aspect we wanted to investigate was the robustness of the proposed model with respect to the amount of feedback received by the agents. We therefore performed experiments with different values of P_{fbk}, while keeping $\omega_A = 1.0$ fixed. Before presenting the results, it is good to recall that when $P_{fbk}= 0$ only $memory_1$ is updated at every language game. In effect this is a CSSL, where the agents do not receive any feedback but have to infer this information by the observation of their contexts. In contrast to earlier CSSL models [10,16,20], the learners additionally receive attentional cues from $strA(o_i)$, which makes the model more similar to the one presented by Gong et al. [5]; and the contexts of speaker and hearer are dissimilar, thus the hearer may not have observed the topic.

The results are reported in Figure 3. It can be noticed that even when there was no feedback ($P_{fbk}= 0$), language developed, though it took more generations to reach an acceptable level of interpretation accuracy than for higher values of P_{fbk} (Fig. 3 left). Interpretation accuracy, although acceptable, was still among the lowest. Production coherence, on the other hand, rose towards a value that is among the highest (Fig. 3 right). This is interesting, since the CSSL generally leads to low levels of coherence [20]. Apparently, the attentional cues provide enough information to allow robust word learning, which is in line with the results of experiment 1 when $\omega_A = 1$. With higher values of P_{fbk} agents receive more feedback and can, therefore, develop a lexicon more easily. Recall that when they receive feedback, they update $memory_2$ as well. Clearly this has a positive effect on speed of learning and on interpretation accuracy. It only has a positive effect on production coherence when $P_{fbk}\geq 0.5$ – i.e. when agents received feedback with a high probability. So, although with a low amount of P_{fbk} accuracy was doing reasonably, coherence remained well behind. This suggests that low amounts of feedback antagonises the attentional cues, since the feedback may

provide different information than the attentional cues, but it has a relatively strong effect on the lexicon development through Eq. 3. When P_{fbk} becomes sufficiently high, the feedback is sufficiently strong to drive a coherent lexicon development. Nevertheless, we can conclude that the model is quite robust to various levels of feedback.

4 Conclusions and Future Work

In this paper a new multi-agent model for the evolution of perceptually grounded lexicons is presented. This model combines the guessing game model [13] with the cross-situational statistical learning model [20] and the introduction of environmental attentional cues similar to the models proposed in [5].

Simulations based on the Talking Heads show that the model is quite robust for different levels of attentional cues set on the objects. However, the simulations show that – in general – the more the attentional cues are used in the interpretation by the hearer, the more the hearer tends to get confused. This is primarily due to the unreliability of the attentional cues, which confuses the hearer. Interestingly, the results improve when the weight for the attentional cues becomes one. In this case, the attentional cues are strong enough to form a beneficial account for the language development.

Another important result is that the model is robust to the enforced dissimilarity of the contexts of agents playing a language game. This is interesting, since it shows that the agents do not require explicit meaning transfer (which is the case whenever feedback is present) while the hearers may not have seen the objects speakers are referring to. Clearly, the results improve when more feedback is present. However, when no feedback is present at all, the results exceed some of the results achieved with infrequent use of corrective feedback, thus showing the robustness of cross-situational statistical learning in combination with using stochastic attentional cues.

Future work should investigate more precisely why the model behaves differently for the different parameter settings. For instance, why do the simulations with higher values of $\omega_A < 1$ or lower values of $P_{fbk} > 0$ perform worse than the cases where $\omega_A = 1$ or $P_{fbk} = 0$? It is also interesting to study the effect of varying P_{fbk} with different values of ω_A. In addition, we intend to incorporate the current model in the recently started New Ties project[2], which aims at developing a benchmark platform for studying the evolution and development of cultural societies in very large multi-agent systems. In this project, we will extend the model such that instead of assigning attentional cues randomly, agents will autonomously estimate (or calculate) these cues as part of a Theory of Mind [19].

[2] http://www.new-ties.org

Acknowledgements

This paper is part of the New Ties project, which is supported by the European Commission Framework 6 Future and Emerging Technologies programme under contract 003752. The authors thank Andrew Smith and three anonymous reviewers for providing invaluable comments on earlier versions of this paper.

References

1. N. Akhtar and L. Montague. Early lexical acquisition: The role of cross-situational learning. *First Language*, 19:347–358, 1999.
2. P. Bloom. *How Children Learn the Meanings of Words*. The MIT Press, Cambridge, MA. and London, UK., 2000.
3. M. M. Chouinard and E. V. Clark. Adult reformulations of child errors as negative evidence. *Journal of Child Language*, 30(3):637–669, 2003.
4. T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13(1):21–7, January 1967.
5. T. Gong, J. Ke, J. W. Minett, and W. S-Y. Wang. A computational framework to simulate the co-evolution of language and social structure. In *ALife 9*, Boston, MA, U.S.A., 2004.
6. J. R. Hurford. Biological evolution of the saussurean sign as a component of the language acquisition device. *Lingua*, 77,2:187–222, 1989.
7. S. Kirby. Spontaneous evolution of linguistic structure: an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation*, 5(2):102–110, 2001.
8. M. Oliphant. The learning barrier: Moving from innate to learned systems of communication. *Adaptive Behavior*, 7 (3-4):371–384, 1999.
9. W. V. O. Quine. *Word and object*. Cambridge University Press, 1960.
10. A. D. M. Smith. Intelligent meaning creation in a clumpy world helps communication. *Artificial Life*, 9(2):559–574, 2003.
11. L. Steels. Emergent adaptive lexicons. In P. Maes, editor, *SAB96*, Cambridge, MA, 1996. MIT Press.
12. L. Steels. The synthetic modeling of language origins. *Evolution of Communication*, 1(1):1–34, 1997.
13. L. Steels, F. Kaplan, A. McIntyre, and J. Van Looveren. Crucial factors in the origins of word-meaning. In A. Wray, editor, *The Transition to Language*, pages 214–217. Oxford University Press, Walton Street, Oxford OX2 6DP, UK, 2002.
14. L. Steels and P. Vogt. Grounding adaptive language games in robotic agents. In C. Husbands and I. Harvey, editors, *Proceedings of the Fourth European Conference on Artificial Life*, Cambridge Ma. and London, 1997. MIT Press.
15. M. Tomasello. *The cultural origins of human cognition*. Harvard University Press, 1999.
16. P. Vogt. Bootstrapping grounded symbols by minimal autonomous robots. *Evolution of Communication*, 4(1):89–118, 2000.
17. P. Vogt. Thsim v3.2: The talking heads simulation tool. In *ECAL03*, pages 535 – 544. Springer-Verlag, 2003.
18. P. Vogt and H. Coumans. Investigating social interaction strategies for bootstrapping lexicon development. *Journal of Artificial Societies and Social Simulation*, 6(1), 1 2003.

19. P. Vogt and F. Divina. Language evolution in large populations of autonomous agents: issues in scaling. In *Proceedings of AISB 2005: Socially inspired computing joint symposium*, pages 80–87, 2005.
20. P. Vogt and A. D. M. Smith. Learning colour words is slow: a cross-situational learning account. *Behavioral and Brain Sciences*, page In press, 2005.

Artificial Life Meets Anthropology: A Case of Aggression in Primitive Societies*

Mikhail S. Burtsev

Keldysh Institute of Applied Mathematics, 4 Miusskaya sq., Moscow RU-125047, Russia
mbur@narod.ru
http://www.keldysh.ru/pages/mrbur-web/

Abstract. One of the greatest challenges in the modern biological and social sciences has been to understand the evolution of altruistic and cooperative behaviors. General outlines of the answer to this puzzle are currently emerging as a result of developments in the evolutionary theories of multilevel selection, cultural group selection, and strong reciprocity. In spite of the progress in theory there is shortage of studies devoted to the connection of theoretical results to the real social systems. This paper presents the model of cooperation which is based on assumptions of heritable markers, constrained resource, and local interactions. Verification of model's predictions with the real data on aggression in archaic egalitarian societies has demonstrated that initial modeling assumptions are acceptable as major factors of social evolution for these societies.

1 Introduction

Many different forms of social organization are found in both historical and contemporary societies, such as kin-groups, bands, tribes, chiefdoms, and states. The question of how these modes of collective action emerge and persist is an important theoretical topic in anthropology, sociology, political science, and history. One influential theory explaining how societies form is the social contract theory, as formulated, for example, by Thomas Hobbes in his work *Leviathan* (1651). Hobbes' main idea is that people submit to the authority of the sovereign, who enforces the social rules, thereby maintaining peace and social stability. The main problem with this theory is that of contract enforcement. Different versions of social contract theory provide their own mechanisms of enforcement, but none of them solve the so called free-rider problem [1].

An alternative view on the emergence of societies is provided by the theories of social evolution. Pioneered in the works of evolutionary biologists [2,3], these theories have been applied to the study of human cooperation and is undergoing intensive development today. Of particular relevance are the theories of multilevel selection [4] and cultural group selection [5]. A series of mathematical models explore how human cooperation can arise [6]. These models are based on the hypotheses of kin and group selection, biased cultural transmission, and perhaps nonrandom (directed) variation. Other models studied the interplay between different kinds of reciprocity and

* This work was supported by the Russian Fund for Basic Research, project 04-01-00510.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 655 – 664, 2005.

punishment and its consequences for the evolution of cooperation [8]. The authors coined the term "strong reciprocity", which refers to the phenomenon that some people have predisposition for altruistic cooperation and altruistic punishment (norm enforcement). Bowles, Gintis and others showed that a small fraction of agents characterized by strong reciprocity could drive the whole population to a cooperative equilibrium. Important results on the evolution of cooperation in the spatially distributed and structured populations were also obtained by Axelrod et al. [8,9]. Finally, it was shown that cooperation could originate in the population of agents with arbitrary tags in the absence of reciprocity [10].

The main tool used today in the field of the evolution of cooperation is game theory. It can be purely analytical game-theoretic models, or agent-based simulations [11]. This theoretical approach yields clear-cut results, but the simple structure of payoffs and a small fixed set of strategies, imposed by investigators, in some cases may be an unrealistic assumption. It is possible to design a much harder test for the theories of social evolution. One such approach, adopted in this paper, is provided by the agent-based evolutionary models in which strategies of agents are not predetermined by researcher but emerge from elementary actions of agents.

The study of cooperation and artificial life are mostly theoretical endeavors: few are grounded with real data. Among them are outstanding examples such as a study of cultural group selection in New Guinea [12] and the simulation of Kayenta Anasazi historical dynamics in Long House Valley [13]. The first study resulted in estimation of cultural change rate. It was argued that significant change of cultural traits under group selection takes from 500 to 1000 years and therefore the more rapid social transformations should be driven by other factors. The results of the second study demonstrated that with the aid of a multi-agent computational model the main features of the history of prehistoric inhabitants of Long House Valley, located in the Black Mesa area of northeastern Arizona (USA) can be closely reproduced. Among these features were population ebb and flow, changing spatial settlement patterns, and eventual rapid decline. The agents in the model were monoagriculturalists, who decide both where to situate their fields and where to locate their settlements.

Filling the gap between the theory and computer modeling on the one hand and objective world on the other is one of the actual tasks of artificial life research. This paper presents an attempt to test predictions generated by rather simple artificial life model of cooperation with the real data. The next section provides description of the model and is followed by a presentation of model's predictions and discussion of their correspondence to the real social systems. The final section is an outline of some conclusions.

2 The Model

The main modeling assumptions were as follows:

- *Evolution.* The strategies in the population evolve through reproduction of agents by the means of mutation and selection.
- *Markers.* Individual markers provide a potential tool for agents to differentiate in-group versus out-group members.
- *Local interactions.* All agents interact locally, as in real social networks.
- *Limited resources.* Agents have limited resources, which serves as a factor of selection in the artificial environment.

The world in the model is a two dimensional closed grid, which forms a torus. There are agents and patches of resource in the world. Only one patch of resource can exist in any cell at a given moment of time, but the number of agents in any cell is unlimited. Patches of resource appear randomly at a constant rate and are uniformly distributed in the space.

An agent can observe its local environment and perform certain actions. The agent is oriented in space and has a field of vision. The field of vision consists of four cells: the cell the agent currently occupies, and the adjacent cells directly to the left, front, and right relative to the orientation of the agent. The agent lives in a discrete time. The agent executes one of seven actions during each time step: rest, consume a resource, turn to the left/right, move forward to the next cell, divide, or fight.

When an agent rests, she changes nothing in the environment. If there is a resource patch in the cell with an agent and she executes the "consume" action, the patch disappears. If the agent divides, an offspring is created and placed in the cell. Each time step before the action is calculated for the given agent one of the other agents in the cell is chosen randomly for potential interaction. The agent can “fight” this chosen one.

Each agent stores a finite amount of resource on which to live. When the agent performs any action, its internal resource decreases. If the agent executes the action "to consume" and there is resource in the cell, the internal resource of the agent increases. When the agent produces offspring, the parent spends some amount of resources in this process and gives half of the rest to the newborn. After executing the "fight" action, the agent takes some amount of resource from the victim. If the internal resource goes to zero, the agent dies.

Each agent has external phenotype that is coded by a vector of integer values (markers). These markers are inherited with mutations by offspring. Thus the Euclidian distance between markers of two agents gives measure of their kinship.

Behavior of the agent is governed by a simple control system. In this system each output associated with a certain action is connected with each input, which is associated with a certain sensory input from environment or internal state of the agent. The control system is a linear system, which is functioning similarly to a feed-forward neural network with no hidden layer. To calculate the output vector $\boldsymbol{O}$ of values, the input vector $\boldsymbol{I}$ should be multiplied by a matrix of weights $\boldsymbol{W}$. Values of $\boldsymbol{W}$ are integers in the range $[-W_{max}, W_{max}]$.

$$O_j = \sum_i w_{ij} I_i \tag{1}$$

At each time step, the agent performs the action associated with the maximum output value.

The input vector $\boldsymbol{I}$ is filled with information about presence of resource and other agents in the field of vision, level of internal resource and Euclidean distance between marker vectors of current agent and its partner for potential interaction.

The weights of the control system are coded in the genome of the agent.

The genome of the agent $\boldsymbol{S}$ consists of three chromosomes $\boldsymbol{S} = (\boldsymbol{B}, \boldsymbol{W}, \boldsymbol{M})$. The first chromosome is the bit string which codes the presence or absence of individual sensory inputs and actions; the second one is the vector of integers which codes the weights of the control system transformation and the third chromosome, also vector of integers, codes the kinship marker of the agent.

If the agent executes the action "divide", its offspring appears. The genome of the offspring is constructed with the aid of the following evolutionary algorithm:

1. for every gene corresponding to the weight of the control system, add a small random integer value uniformly distributed on the interval $[-p_w, p_w]$, where p_w is mutation intensity;
2. with a small probability p_b, change each bit for the presence of sensory input or action;
3. for every gene corresponding to the kinship marker, add a small random integer value uniformly distributed on the interval $[-p_m; p_m]$, where p_m is the mutation intensity of the marker.

More details on the implementation of the model could be found in [14].

3 Results and Discussion

The model described in the previous section lacks mechanisms for complex social interactions thus the simulation results are not applicable to just any complex society. But it is reasonable to conjecture that modeling assumptions (see the beginning of the previous section) hold for the archaic egalitarian societies such as communities of hunter-gatherers and primitive agriculturalists. This section is devoted to testing this hypothesis.

One of the largest domains in the area of ethnographic, anthropological, and cross-cultural studies of primitive societies is committed to the research on interrelations of resources availability, aggression, and population pressure [15-22]. So it can provide us with a variety of theories to compare and with data on the real societies to verify predictions of the simulations.

Below the agent from the model will be treated as a community of hunter-gatherers (a band) or primitive agriculturalists. In egalitarian societies, a community consisting of few nuclear or one extended family behaves like autonomous entity [23]. Members of community move, settle, and fight together. It is assumed that internal resource of an agent (from the model) corresponds to the human resource of community (its size). These assumptions allow us to grasp the following features of primitive societies in the model.

1. *Capturing enemies.* For primitive societies it was a common practice that captured during an attack men were used as slaves, women as wives and children were adopted [19,20]. In the model when one agent fights another agent the former captures some amount of internal resource of the latter.
2. *Evolution is based on already obtained adaptation.* If one accepts that prehistoric humans evolved pre-adaptations for hunting and gathering in small bands ("tribal social instincts" hypothesis [24]) it should be expected that development of more complex social organization will be based on and constrained by this pre-adaptations. This "tribal social instincts" hypothesis is manifested in the model as a limitation of the capacity of agent's internal resource. In other words it is assumed that a community exists as stable social entity if its size is under some threshold (maximal size of a band).

The evolution of cooperation in the model is based on the presence of phenotypic markers. The model marker of an agent is inherited in the same manner as a strategy of its behavior. Therefore agents with similar markers will have similar behavior and it is reasonable to think of ability of agents to differentiate each other markers in the model as ability of actors in the real social systems to perceive common descent or cultural markers.
Two measures were introduced to estimate an aggression level in the model.

1. *Frequency of execution of the "fight" action in a population.*
2. *Frequency of aggressive agents in a population.* Here an aggressive agent is an agent which can potentially fight other agent at the cell where it is situated. Every agent in a population was tested with a fixed set of most frequent alternatives of interactions (similar vs. different marker, low vs. high internal resource) and if in any situation an agent has performed a "fight" action it was treated as "aggressive".

As it was mentioned in the beginning of this section a large body of research on aggression in primitive societies is devoted to the study of dependence of aggression level and population density on variation in resource supply. Resources available to community in primitive societies are generally dependent on ecological conditions and on level of resources extraction technology. In the model bundles of resource which an agent can consume appears in every cell of environment with some constant probability. In a series of simulations amount of a resource in a bundle C was varied in one order of magnitude. The range of variance in amount of resource in a bundle was set in a way that for smallest values it was insufficient for agent survival without them moving out of cell. At highest values of resource supply allowed survival of few agents in one cell. As a result the simulations with low resource supply correspond to hunter-gatherers in poor ecological conditions and the simulations with high resource supply to primitive agriculturalists and hunter-gatherers with rich resource base.

The dependence of population density on an amount of a resource in a bundle C breaks on three parts (see fig. 1a). When resource supply is insufficient to support survival of one agent in the cell $C < C_1$ population density reaches maximum for the given value of the C. If resource base is sufficient for survival of one agent in the cell but not two $C_1 \leq C < C_2$ then the number of agents per cell does not depend on C and equals 1. In this case every cell is usually occupied by only one agent which prefers don't move. If there is other agent in the cell then the dominant strategy to escape the cell in the case of small distance between marker-vectors and fight in opposite. For the $C \geq C_2$ there is no dominant strategy in a population. The model demonstrates complex interplay of mixture of cooperative and non-cooperative predator and prey strategies which results in oscillation of population density.

Applying the two measures of aggression—proposed above—to the simulation results gives dependencies which are presented on the figures 1b and 1c. Predictions of the model can be summarized as follows.

1. In the condition of poor resource base ($C < C_1$), a portion of aggressive agents and a frequency of acts of aggression should grow with increase of resource supply.
2. At intermediate resource supply ($C_1 \leq C < C_2$), almost every agent should be able to fight (fig. 1c) but frequency of actually performed aggressive actions is very low (fig. 1b).
3. In the case of rich resources ($C > C_2$), both amounts of aggressive agents and acts of aggressions should increase as supply rises.

Modern anthropology suggests that for primitive societies a general tendency is an increase of aggression with an enhancement of environmental conditions [16,17,21,22]. Simulation results fit this tendency for the ranges of poor ($C < C_1$) and rich ($C > C_2$) resource base (see fig. 1b and fig. 1c). But for the intermediate values of resource supply ($C_1 \leq C < C_2$) it looks like that simulations contradict anthropological theory. On the other hand data about four Kalahari Bushmen groups which were provided in the influential work of Cashdan [17] shows exception from the general theory. Among these four groups, !Ko, G/wi, Nharo, and !Kung, the first lived in the poorest ecological conditions and the last in the best but the !Ko demonstrated the most territorial and aggressive behavior and the !Kung were looked the most peaceful.

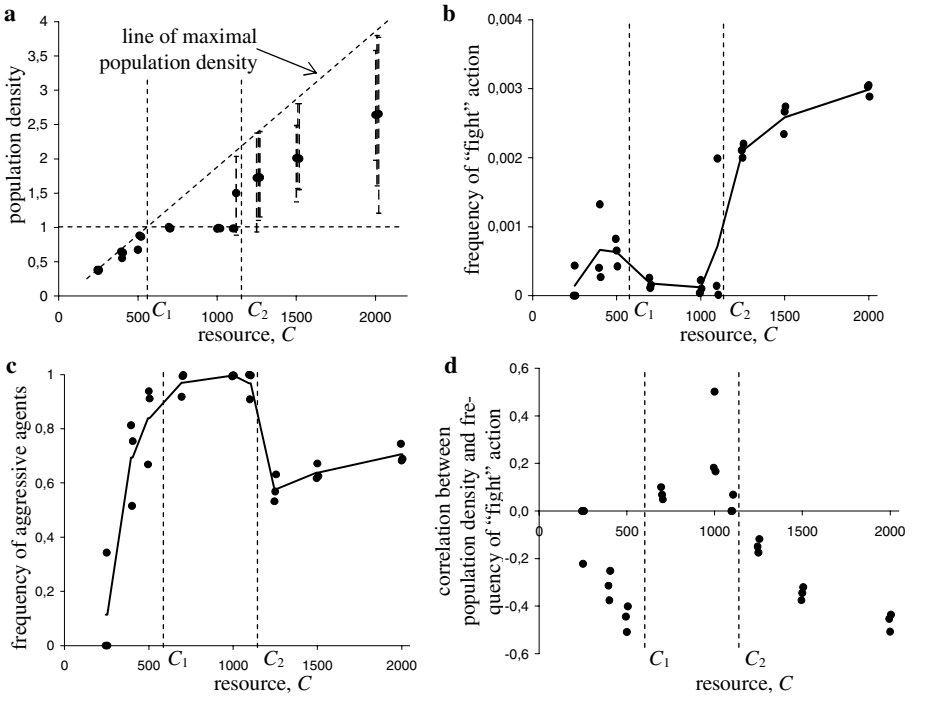


Fig. 1. Dependence of population density (a), frequency of "fight" action (b), frequency of aggressive agents (c), and correlation between population density and frequency of "fight" action (d) on resource supply. Dots correspond to the values for different simulation runs; solid lines connect averages.

If one tries to align the data from Cashdan [17] with the simulations results it is reasonable to consider resource supply of the !Ko as falling in the range $C < C_1$ and resource supply of the !Kung in the range $C_1 \leq C < C_2$. In this case the model predicts a higher aggression rates for the !Ko and a lower for the !Kung (fig. 1b). So the model's prediction matches the data in this case. Another prediction of the model requires the !Kung to be ready to perform violent act at any moment (fig. 1c). The !Kung are hunting with bow and poisoned arrows, so there are no technical problems for the !Kung bushman to kill another man. Moreover the !Kung have norms prescribing circumstances in which one !Kung is allowed to kill another one. For example, the bushman who finds a hive obtains rights on the honey from that hive. The owner of a hive is allowed to kill anybody who attempts to take honey from the hive without permission. Such features of the !Kung bushmen as low level of observable aggression and high potential for producing violent acts fit surprisingly the second prediction form the list above. The contradiction between the actual and potential aggression is not unique for the !Kung; a similar pattern can be found among Australian's aboriginals. Aboriginals from the Western Desert have ecological conditions similar to the !Kung. These aboriginals demonstrate low level of violence as the !Kung do but have institutes of socialization for aggression such as ritual fights and formation of secret groups of avengers, etc [21,22].

The Malthusian suggestion that population pressure should lead to war is commonly accepted in current anthropology for the case of preindustrial stateless societies [22]. Recent developments in the modeling of warfare in primitive agricultural societies are based on the approaches of population dynamics [26] and also consider population density as the major determinant positively affecting the level of warfare. The straight Malthusian approach predicts positive correlation between population density and frequency of warfare. The models of Turchin and Korotaev [26] give a weak negative correlation. Correlations between population densities and frequency of "fight" actions in the population for the simulations with the artificial life model are given on the figure 1d.

A cross-cultural test has been completed to compare all three predictions. As the source of data on real societies a Standard Cross-Cultural Sample database [27] was used. The correlations between density of population and few internal warfare variables were calculated for the societies with low level of political integration and extensive agriculture as subsistence technology. The results of cross-cultural test are presented in the table 1.

The results presented in the table 1 are in agreement with results of similar analyses provided in [22]. The analysis reveals rather strong negative correlation between variables "Density of Population" and "Frequency of Intercommunity Armed Conflict" $r = -0{,}489$, $p = 0{,}046$. Societies with extensive agriculture correspond to the simulated populations with rich resource supply ($C > C_2$). Simulation results for the highest value of resource (fig. 1d, $C = 2000$) give the closest match to the data. The pure Malthusian approach and the models of Turchin and Korotaev are not supported by data.

In addition to the population pressure one more factor believed to affect warfare is predictability of resources. Embers [28-30] showed that resource problems, particularly those created by unpredictable weather or pest disasters strongly predict warfare frequency (for direct archaeological evidence on unpredictable resource fluctuations

as a major factor of warfare frequency see, e.g., [31,32]). Multivariate analyses for nonstate societies gave standardized coefficient of $r = 0{,}631$ ($p < 0.001$, one tail) for natural disasters as predictor of warfare ([29] p. 254). Furthermore, the correlation between the presence of unpredictable natural disasters destroying food supplies and warfare frequency has turned out to be stronger than the one attested for more than a dozen various warfare frequency factors tested by the Embers.

Table 1. The results of cross-cultural test[1] (data for the test are taken from a Standard Cross-Cultural Sample database [27])

	"Density of Population" v156	
Name of variable and its number in Standard Cross-Cultural Sample [27]	*r*	*p**
"Frequency of Intercommunity Armed Conflict" v693	**-0,489**	**0,046**
"Frequency of Violent Conflict Between Groups within Local Communities" v1750	-0,230	0,375
"Frequency of Violent Conflict Involving at Least One Local Community" v1758	-0,208	0,408

*A correlation is significant if $p < 0{,}05$.

Table 2. Two measures of aggression for the high and low predictability of resources in the model

	case 1	case 2	case 3	average
high predictability of resources				
Frequency of execution of the "fight" action in a population.	0,000686	0,000807	0,000791	**0,000761**
Frequency of aggressive agents in a population.	0,119	0,176	0,172	**0,156**
low predictability of resources				
Frequency of execution of the "fight" action in a population.	0,00234	0,00267	0,00274	**0,00258**
Frequency of aggressive agents in a population.	0,617	0,672	0,625	**0,638**

To test if the model could grasp this phenomenon, two series of simulations were performed [33]. They differ in amount of resources in a patch and frequency of patch appearance. For the first series, the frequency of resource appearance was ten times greater than for the second, but amount of resources in a patch was ten times smaller than for the second. So, for both cases total amount of resources which could be collected by agent during given period of time was equal, but the probability (and, hence,

[1] Only the cases for which the following conditions hold were selected for the test. 1. A variable "Political Integration" (v157) should have one of the values "None", "Autonomous local communities", or "1 level above community". 2. A variable "Intensity of Cultivation" (v232) should have the value "Extensive or shifting agriculture, long fallow, and new fields cleared annually".

predictability) of obtaining a single portion of resource for the first series was ten times greater than for the second.

As shown in table 2 measures of aggression proposed above drop in almost three times for simulations with high predictability of resources with respect to low predictability. So the model is deemed to have passed in the third test as well.

4 Conclusion

The artificial life model of evolution of cooperation—based on assumptions of heritable markers, constrained resource, and local interactions—demonstrates surprising fit to some features of real social systems. The model captures a general trend of increasing of the aggression level with a rising resource supply in primitive societies but grasps also some exceptions such as a case of !Ko and !Kung in Kalahari desert which demonstrates reverse interdependence between resource base and aggression. At some level of resources in environment, the model predicts mismatch between levels of actually manifested aggression and the propensity to perform violent acts. This prediction finds support in the behavior of !Kung bushmen and aboriginals of Western Desert of Australia. The correlation between population density and frequency of fight action for the case of rich resources in the model is similar to the analogous correlation extracted from ethnographic database. Finally, impact of resource predictability on internal warfare observed for real societies is correctly replicated in the model's behavior. All this allows us to consider that initial modeling assumptions are acceptable as major factors of social evolution in archaic egalitarian societies.

Acknowledgements

Thanks to Peter Turchin who fed me with the idea to model cooperation, Daria Khaltourina for the introduction to the cross-cultural method and discussion of modeling assumptions, Andrey Korotayev for the idea to study how aggression in the model depends on the resource predictability. I deeply acknowledge the anonymous reviewers for their suggestions and comments.

References

1. *Kraus J.S.* The Limits of Hobbesian Contractarianism. – Cambridge: Cambridge University Press, 1993.
2. *Hamilton W.D.* Innate social aptitudes of man: An approach from evolutionary genetics / In: Biosocial Anthropology, ed. R. Fox, p.115–132. New York: Wiley, 1975.
3. *Wilson E.O.* Sociobiology: The new synthesis. – Cambridge: Belknap Press, 1975.
4. *Sober E., Wilson D.S.* Unto Others: The Evolution and Psychology of Unselfish Behavior. – Cambridge: Harvard University Press, 1998
5. *Richerson P.J., Boyd, R.* Not By Genes Alone: How Culture Transformed Human Evolution. – Chicago: University of Chicago Press, 2005.
6. *Boyd R., Richerson, P.J.* The Origin and Evolution of Cultures. – Oxford: Oxford University Press, 2005.

7. *Bowles S., Gintis H.* The evolution of strong reciprocity: cooperation in heterogeneous populations // Theoretical Population Biology V.65, p.17-28, 2004.
8. *Axelrod R.* The evolution of cooperation. – New York: Basic Books, 1984.
9. *Cohen M.D., Riolo R L. Axelrod R.* The role of social structure in the maintenance of cooperative regimes // Rationality Soc. V.13, p.5-32, 2001.
10. *Riolo R., Cohen M.D., Axelrod R.* Evolution of cooperation without reciprocity // Nature V.414, p.441–443, 2001.
11. *Epstein J.M., Axtell R.L.* Growing artificial societies: Social science from the bottom up. – MA: MIT Press, 1996.
12. *Soltis J., Boyd R., Richerson P.J.* Can group-functional behaviors evolve by cultural group selection? An empirical test // Current Anthropology V.36, p.437-94, 1995.
13. *Gumerman G.J., Swedlund A.C., Dean J.S., Epstein J.M.* The Evolution of Social Behavior in the Prehistoric American Southwest // Artificial Life V.9, p.435–444, 2003.
14. *Burtsev M.S.* Tracking the Trajectories of evolution // Artificial Life V.10, p.397-411, 2004.
15. *Vayda A.P.* Warfare in Ecological Perspective // Annual Review of Ecology and Systematics, V.5, p.183-193, 1974.
16. *Dyson-Hudson R., Smith E.A.* Human territoriality: An Ecological Reassessment // American Anthropologist V.10, p.21-41, 1978.
17. *Cashdan E.* Territoriality among Human Foragers: Ecological Models and an Application to Four Bushman Groups // Current Anthropology V.24, N1, p.47-55, 1983.
18. *Ember C.R., Ember M.* War, Socialization and Interpersonal Violence: A Cross-Cultural Study // The Journal of Conflict Resolution V.38, N4, p.620-646, 1994.
19. *Shnirelman V.A.* The roots of war and peace. War and Peace in the Early History of Humankind, V.1. – Moscow: IEA RAS, 1994. (In Russian)
20. *Otterbein K.F.* Killing of Captured Enemies: A Cross-cultural Study // Current Anthropology V.44, N3, p.439-443, 2000.
21. *Kazankov A.A.* Factors of Intercommunity Aggression In Archaic Societies: Hunter-Gatherers of the Semi-Arid Zones. – Moscow: IA RAS, 2002. (In Russian)
22. *Nolan P.D.* Toward an Ecological-Evolutionary Theory of the Incidence of Warfare in Preindustrial Societies // Sociological Theory V.21, N1, p.18-30, 2003.
23. *Kabo V.R.* Primitive Preagricultural Community. – Moscow: Nauka, 1986. (In Russian)
24. *Richerson P.J., Boyd R., Henrich J.* The Cultural Evolution of Human Cooperation // The Genetic and Cultural Evolution of Cooperation, P. Hammerstein ed., p.357–388. Cambridge MA: MIT Press, 2003.
25. *Terry M.* War of the Warramullas. – Adelaide: Rigby, 1974.
26. *Turchin P., Korotayev A.* Population Dynamics and Internal Warfare: a Reconsideration // *submitted to* American Anthropologist.
27. Standard Cross-Cultural Sample Variables (Edited by William T. Divale, Daria Khaltourina and Andrey Korotayev) // World Cultures V.13, 2002.
28. *Ember M.* Statistical evidence for an ecological explanation of warfare // American Anthropologist V.84, p.645-649, 1982.
29. *Ember C.R., Ember M.* Resource Unpredictability, Mistrust, and War: A Cross-Cultural Study // Journal of Conflict Resolution V.36, p.242-262, 1992.
30. *Shankman P.* Culture contact, cultural ecology, and Dani warfare // Man V.26 p.299-321, 1991.
31. *Kang B.W* A Reconsideration of Population Pressure and Warfare: A Prehistoric Korean Case // Current Anthropology, V.41, p.873–881, 2000.
32. *Lekson S.H.* War in the Southwest, War in the World // American Antiquity V.67, p.607-624, 2002.
33. *Burtsev M.S., Korotaev A.V.* An Evolutionary Agent-Based Model of Pre-State Warfare Patterns: Cross-Cultural Tests // submitted to World Cultures.

Emergence of Structure and Stability in the Prisoner's Dilemma on Networks

Leslie Luthi, Mario Giacobini, and Marco Tomassini

Information Systems Department, University of Lausanne, Switzerland
{leslie.luthi, mario.giacobini, marco.tomassini}@unil.ch

Abstract. We study a population of individuals playing the prisoner's dilemma game. Individual strategies are invariable but the network of relationships between players is allowed to change over time following simple rules based on the players' degree of satisfaction. In the long run, cooperators tend to cluster together in order to maintain a high average payoff and to protect themselves from exploiting defectors. We investigated both synchronous and asynchronous network dynamics, observing that asynchronous update leads to more stable states, and is more tolerant to various kinds of perturbations in the system.

1 Introduction

The *Prisoner's Dilemma* (PD) game is a methaphor for analyzing conflicting situations that arise in the economy and in society in general (see Axelrod's book [1]). It has fascinated researchers because it is an interaction where the individual rational pursuit of self-interest produces a collective result that is self-defeating. The following payoff matrix represents the prisoner's dilemma game as a two-person game in normal its form [4]:

	C	D
C	(R,R)	(S,T)
D	(T,S)	(P,P)

In this matrix, C stands for *cooperation* and D for *defection*. R stands for the *reward* the two players receive if they both cooperate, P is the *punishment* for bilateral defection, and T is the *temptation*, i.e. the payoff that a player receives if she defects, while the other cooperates. In this latter case, the cooperator gets the *sucker's* payoff S. The payoff values are submitted to the following constraints: $T > R > P > S$ and $R > (T + S)/2$.

This game has a unique Nash equilibrium, (P,P), and thus, in a one-shot play of the game, the rational outcome is for both players to play D in spite of the fact that both players would be better off cooperating, with a payoff (R,R).

If the PD game is iterated a known finite number of times, the result doesn't change, and steady defection of the two players is the rational outcome of each encounter in the sequence. However, when the game is iterated an *indefinite* number of times, strategies that allow cooperation to emerge and persist are

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 665–674, 2005.

possible, as described by Axelrod [1,3]. This result could lend some justification to the commonly observed fact that cooperation does appear in society, in spite of individual greed. We will only deal with the one-shot case in the rest of the paper.

Considering now not just two players but rather a population of N players, *evolutionary game theory* [8] prescribes that defection is the evolutionarily stable strategy of the population, given memoryless players. However, in 1992, Nowak and May [7,6] showed that cooperation in the population is sustainable under certain conditions even in the one-shot game, provided that the population of players has a lattice *spatial structure*. Nevertheless, many real conflicting situations in society are not well described by a fixed geographical position of the players. In economy, for instance, markets and relations between firms are not limited by geographical distance in this era of fast global communication. The same can be said of many social and political interactions where relationships may change over time. Thus, it becomes of interest to study how the interaction network influences the global outcome, and how this same relational structure may evolve under the pressure of the player's strategic interactions. The PD is an excellent way of studying such an evolution in a simplified and understandable environment.

Recently, Zimmermann *et al.* [9] have published a study in which both the strategies of players – C or D – and the network of players' relationships may change and adapt during time. They use the same synchronous strategy evolution as Nowak and May [7]. Players are *satisfied* if their payoff is the highest among the neighbors, otherwise they are *unsatisfied*. Only unsatisfied D-players can then break a link to another D-player with a certain probability and rewire it randomly. Zimmermann *et al.* find that, for some value of the parameters, the network of players self-organizes to stable cooperative states.

Here we follow a similar idea but we assume a population of players each of which has an unchanging strategy C or D. Thus, we concentrate on the purely topological aspects i.e., the evolution of the network of relationships among the players. Although the assumption of unchanging strategy may seem unrealistic, this is not necessarily so. Indeed, there are many situations in which the player has little or no choice of alternative strategies, either because of insufficient knowledge or because of external social pressure. With respect to [9], our network update rules are different, as explained in the following section. We study both synchronous and asynchronous update policies and compare the results on deterministic as well as perturbed systems. A preliminary account of the noiseless model appears in [5].

2 The Model

We consider a population of N individuals all playing the prisoner's dilemma. The population can be subdivided into the subset of the cooperators, E_C, and the one comprising the defectors, E_D. Initially, there are no links between the N players in the population. At a given time t, an individual i interacts exclusively

with a subset of the entire population known as its *neighbors* and denoted by $V_i(t)$ with the condition $i \notin V_i(t)$, $\forall t$. An individual i does not necessarily have neighbors in which case $V_i = \emptyset$. An interaction of an individual i with one of its neighbors j is represented by an undirected link so that $i \in V_j(t) \Rightarrow j \in V_i(t)$, $\forall t$. The values in the payoff matrix $S = 0$, $P = 0.1$, $R = 1$, $T = 1.39$ have been chosen to be in accordance with the $T > R > P > S$ and $R > (T + S)/2$ relationships. Furthermore, $T = 1.39$ was chosen due to the interesting results obtained in previous works as to the persistance of C and D together for values of T in between 1.2 and 1.6 [6].

2.1 Average Payoff and Notion of Satisfaction

Let s_i be the strategy of an individual i, with $s_i = 0$ for a defector (D) and $s_i = 1$ for a cooperator (C). Furthermore, let us denote with $\Pi_i(t)$ the average payoff of the individual i at a given time step t. This gives us the following equation:

$$\Pi_i(t) = \begin{cases} -1 & \text{if } V_i = \emptyset, \\ \frac{s_i(\gamma_i(t)R+\delta_i(t)S)+(1-s_i)(\gamma_i(t)T+\delta_i(t)P)}{|V_i(t)|} & \text{otherwise.} \end{cases} \quad (1)$$

where $\gamma_i(t)$ (resp. $\delta_i(t)$) is the number of cooperators (resp. defectors) $\in V_i$ at the given time step t. $\Pi_i(t)$ has a negative value when player i is isolated to distinguish this case, where the player has to randomly choose another player in the population to be its neighbor, from the case where the player has a 0 payoff and must thus rewire one of its links. Once $\Pi_i(t)$ is defined, we can introduce the *satisfaction threshold* Θ_i of an individual i as:

$$\Theta_i = s_i(S + \sigma(R - S)) + (1 - s_i)(P + \sigma(T - P)) \quad (2)$$

where $0 \leq \sigma \leq 1$ is called the *satisfaction degree*.
An individual i is said to be *satisfied* iff $\Pi_i \geq \Theta_i$. The *satisfaction degree* characterizes the minimum percentage of cooperators an individual should have among its neighbors in order to be satisfied. The two limit cases are $\sigma = 0$, which means that an individual i will always be satisfied except if it has no neighbors ($V_i = \emptyset$), and $\sigma = 1$ implying that i will be satisfied iff its neighborhood is composed solely of cooperators (i.e. $\Pi_i = \Theta_i = s_iR + (1 - s_i)T$). Our notion of a satisfied individual differs from that of [9] where a player is satisfied only if it has the highest payoff among its neighbors. In a real-world situation, one doesn't necessarily desire to be "the best", and, moreover, a player might find it impossible to know the neighbors' payoffs. Furthermore, we find it important to work with an average payoff instead of the accumulated one considered in [9].

2.2 The Rewiring Rules and the Time Evolution

Let i be a player and t_n the n_{th} time step. Its interaction with other players evolves in time according to the following basic rules:

- *if* $V_i(t_n) = \emptyset$: i is unsatisfied and must choose an individual j uniformly at random in the population to be i's new neighbor, i.e. $j \in V_i(t_{n+1})$.
- *else if* $\Pi_i(t_n) < \Theta_i$: i is unsatified and must hence pick randomly one of its D-neighbors and replace it with a randomly chosen individual j satisfying $j \notin V_i(t_n)$. If such a j does not exist, nothing is done and i must try to bear with its unsatisfaction. Notice that an unsatisfied individual i with $V_i \neq \emptyset$ necessarily has a D-neighbor since only C-neighbors contribute to i's satisfaction.
- *otherwise*: i is satisfied with its situation and will thus continue to play against exactly the same individuals at time step t_{n+1} unless, independent of i, one of its neighbors decides to cut off its link with i or an outsider inserts itself into i's neighborhood.

Finally, let us stress the fact that unlike the model in [9], *cooperators, as well as defectors*, have the possibility to attempt a change in their neighborhood.

Synchronous Time Evolution. In the *synchronous* update, every individual i starts by playing the PD game with its neighbors and uses the outcome to calculate its average payoff $\Pi_i(t_n)$. Next, all the unsatisfied players simultaneously modify their neighborhood according to the previously mentioned rules (2.2). Possible collisions are resolved by using a temporary network data structure.

Asynchronous Time Evolution. The hypothesis of a global clock, which is a necessary requirement for synchronous dynamics, could be unrealistic in a social setting, since information travels at finite speed. Thus, having the whole population update its state all at once is only an idealization. Hubermann and Glance [2] elaborate on this point. A different point of view is discussed in [6] where the authors state that there is not much difference at the macroscopic, population level. Here we have decided to study both update models, in order to gain further insight on the corresponding dynamical processes.

For the *asynchronous* case, while other policies could be used, we chose the customary *independent random ordering* of updates in time which is a close approximation of a Poisson process. The time t needed to update the whole population is subdivided into a sequence $(u_1, u_2, \ldots, u_N)$ of *update steps*. During an update step u_k, an individual i is randomly picked, $\Pi_i(u_k)$ is calculated, and the appropriate rules (2.2) are used to immediately adapt its neighborhood if unsatisfied. Note that in the rules 2.2, $V_i(t_{n+1})$ becomes $V_i(u_{k+1})$, $k = 1, 2, \ldots, N$. This process is iterated N times with replacement (where N is the population size). Note that this is only one of many possible sequential update policies, but it is a reasonable one in our case.

3 Simulation Results and Analysis

$0.0 \leq \sigma \leq 0.9$. In order to analyze the influence of the satisfaction degree on the network of players and compare the results between the synchronous and asynchronous updates, we varied σ from 0.0 to 0.9 by steps of 0.1. For each of

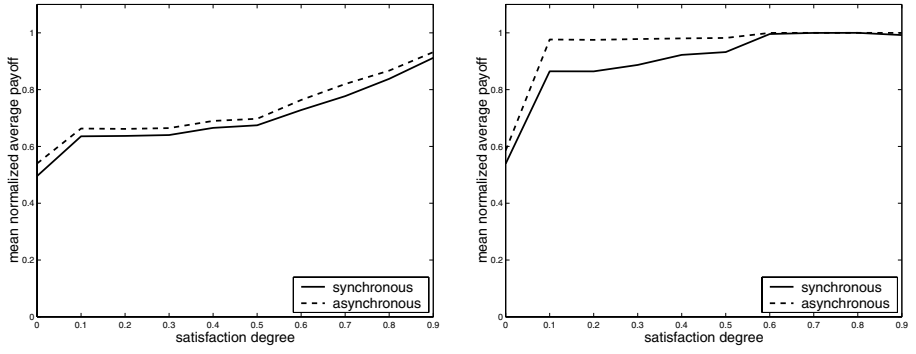


Fig. 1. The mean normalized average payoff $\overline{\Pi}_C$ (a) and $\overline{\Pi}_D$ (b) after the final time step, averaged over 20 runs. Synchronous versus asynchronous updating.

these values, two series of 20 runs of 400 time steps each were executed on a population of 1600 players (50%C – 50%D), one series per update policy.

Among the several quantities that were observed at the end of each run, we studied in particular the *mean normalized average payoff* $\overline{\Pi}_C(t)$ and $\overline{\Pi}_D(t)$ of cooperators and defectors respectively, defined as:

$$\overline{\Pi}_C(t) = \frac{\sum_{i \in E_C} (\Pi_i(t) - S)}{(R - S)\,|E_C|}, \qquad \overline{\Pi}_D(t) = \frac{\sum_{i \in E_D} (\Pi_i(t) - P)}{(T - P)\,|E_D|} \tag{3}$$

As the satisfaction degree increases, the mean average payoff of the cooperators seems to clearly tend to the C maximum payoff R (Fig. 1(a)). Moreover, both synchronous and asynchronous lead to the same behavior. Unlike $\overline{\Pi}_C$, $\overline{\Pi}_D$ differs a little from synchronous to asynchronous. Fig. 1(b) shows that when using an asynchronous update, the defectors globally attain a payoff close to the their maximum possible ($\overline{\Pi}_D > 0.9$) for already very small values of σ ($\sigma = 0.1$) whereas in the synchronous case, $\overline{\Pi}_D$ greater than 0.9 are reached only for $\sigma > 0.5$. Nevertheless, if we ignore transients, i.e. the initial phases of the evolution, the long-term trend is identical for the two types of updates. The correlation between the increase of σ and that of $\overline{\Pi}_C$ and $\overline{\Pi}_D$ is explained easily by the fact that the higher the satisfaction degree, the more demanding the players become of their neighbors. For example, $\sigma = 0.8$ implies that in order for an agent to be satisfied, its neighborhood must be composed of at least 80% of cooperators.

$\sigma = 1.0$. Do the previous tendencies hold true for the limit case of $\sigma = 1$ where all the players, whether they are cooperators or defectors, are unsatisified as long as a defector is found among its neighbors? To answer this question, since the simulations are time-consuming, the runs were increased to 1000 time steps each and the size of the population was reduced to 40 individuals (20 C and 20 D).

The time series of the mean normalized average payoffs $\overline{\Pi}_C$ and $\overline{\Pi}_D$ are shown in Fig. 2.

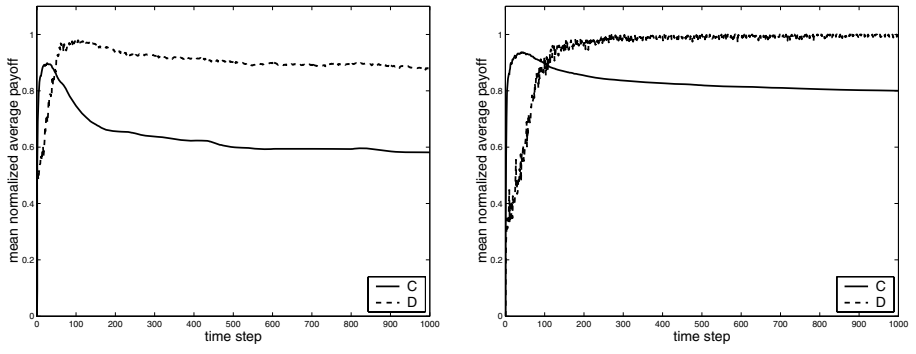


Fig. 2. Evolution of $\overline{\Pi}_C$ and $\overline{\Pi}_D$ averaged over 20 runs; (a) using synchronous update, (b) using asynchronous update

An interesting point to notice is that when using an asynchronous update, all the D-players are able to reach and maintain their maximum payoff. The small fluctations that occur after system stability (generally reached around time step 150) are due to defectors getting totally disconnected from their neighbors (which were of course all unsatisfied cooperators). However, the lone defectors reattain their maximum payoff by finding a new C-neighbor usually in a matter of one to two time steps. In the synchronous case, the defectors on average seem to tend to their maximum payoff in the first 100 time steps, after which $\overline{\Pi}_D$ gently decreases. This is also true concerning the payoff of the cooperators which diminishes at the the same speed as $\overline{\Pi}_D$ does.

Fig. 2 unfortunately does not give a detailed picture of the rewiring at hand and the underlying organization, since once again it only shows an average of the 20 runs. In order to have a better understanding, we must have a look at a few typical runs and study not only $\overline{\Pi}_C$ and $\overline{\Pi}_D$, but also the evolution of the different degrees of the network.

The first thing to be observed is the cooperators forming a complete graph whether the update is synchronous or asynchronous (see Fig. 3 where, after a transient period, the average degree of cooperators becomes 19, which is the maximum possible degree given that there are 20 cooperators in the population). This is a direct consequence of the pressure $\sigma = 1$ exerts on the C-players forcing them to continue changing neighbors for as long as they are connected to defectors. Since defectors, on the other hand, continously seek to have interactions with them, the C-players inevitably end up forming a huge cluster where they are all linked to one another.

Secondly, as mentioned above, when using an asynchronous update, all the defectors reach their maximum payoff and maintain it (Fig. 2(b)). Now, looking at Fig. 4(b) showing the mean normalized average payoffs, we see that the local drops of $\overline{\Pi}_D$ are due to defectors finding themselves cut off from the cluster of cooperators and having thus a momentarily negative average payoff (Eq. 1). These defectors will then randomly create a new link, rewiring it until it

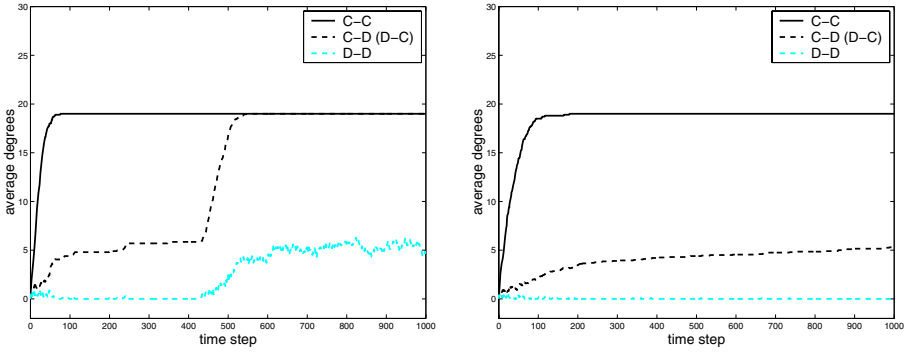


Fig. 3. Average degrees of a typical run; update: (a) synchronous, (b) asynchronous; population size: 40, $\sigma = 1$. C-D and D-C links are the same since there are as many cooperators as defectors.

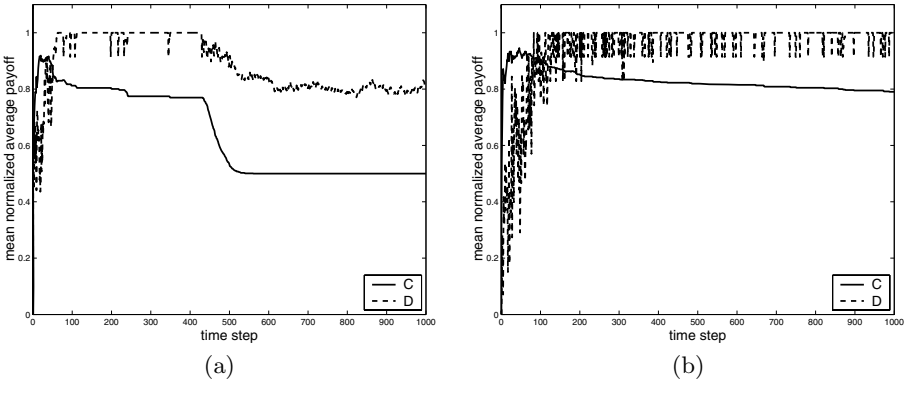


Fig. 4. Time evolution of the mean averaged payoffs $\overline{\Pi}_C$ and $\overline{\Pi}_D$ during a typical run; update: (a) synchronous, (b) asynchronous; population size: 40, $\sigma = 1$

eventually reconnects them to a cooperator. Only then will the system reach once again a stable state. Further details on this behavior can be found in [5].

When using a synchronous update, things are not as simple. Fig. 3(a) and Fig. 4(a) show that although the system reaches a stable state, the latter is quite fragile and is not safe from collapsing to attain another type of stable state.

The downfall of both $\overline{\Pi}_C$ and $\overline{\Pi}_D$ begins with a D-agent being isolated. Once this occurs, there is a fifty-fifty chance that the latter will generate a link coupling it with another previously satisfied D-player. At the next time step, both these unsatisfied defectors will attempt to rewire the same link, thus creating two new links for only one "disappearing". If these two defectors are associated in turn to other defectors, there's the risk that this process will snowball, producing more and more links connecting D-agents to other individuals. This explains why, when using a synchronous update, a majority of the runs present sooner or

later a rise of the different average degrees accompanied by a drop of both $\overline{\Pi}_C$ and $\overline{\Pi}_D$. The system usually reattains stability once the $C - D$ average degree reaches $\frac{N}{2} - 1$ where N is the size of the population (see Fig. 3(a)).

On the same figure, the fact that the D-D average degree does not increase past a certain level — about six in the figure — is due to the fact that the higher the D-D average degree, the higher the probability of two unsatisfied D agents breaking a link with other D players, and trying both to connect to one another. This results in two D-D links disappearing for only one new D-D link created.

To ascertain that the results obtained with 40 individuals are also valid for bigger size populations, a few very long runs (12000 time steps) were executed on the initial size of 1600 players. These runs — not shown here to save space — qualitatively confirm the dynamics observed in the small size population.

4 System Stability Under Noise

In order to study the resilience of the system in the case of $\sigma = 1$, two kinds of perturbations were applied. The first one takes place only once the stability is attained i.e., at time step 1001. At that point, a certain percentage of cooperators becomes defectors for a lapse of 10 time steps, after which they regain their initial strategy. The second one concerns an error an agent makes when deciding to rewire one of its links. Instead of breaking a link with one of its D-neighbors, a player might go awry with a certain probability and break up with a randomly chosen C-neighbor. In both cases, we only show the fluctuations over time of $\overline{\Pi}_D$ averaged over 20 runs. This characteristic, as well as the sundry average degrees, are where we find the most significant differences between the two types of updates.

Strategy Noise. The perturbations on the players' strategy are of two types: one where the changes are applied to cooperators alone and one where only the defectors are allowed to turn into cooperators. The latter case is not of great interest. Indeed, at system stability, when using an asynchronous update, Ds are merely satellites around the complete cluster of Cs with links solely with the latter. Therefore, a defector who turns into a cooperator will immediately be satisfied and there will be practically no perturbation on the system between the moment an error is introduced and the moment the system recovers its normal state. In the case of synchronous updating, according to the discussion in section 3, we either find ourselves in a situation similar to the asynchronous one previously discussed, or the system has already given way to the sudden increase of D-D links (see Fig. 3). In the latter case, a previously D-agent connected to practically all the cooperators, will form, as a cooperator, a complete cluster with his fellow cooperators in a matter of a few time steps. Once the initial strategies of the players are reestablished, the system will immediately recover its original state with simply a higher average degree.

For the case of a C turning into a D, Fig. 5 indicates that, when using an asynchronous update, the system regains its previous level of defector payoff, even

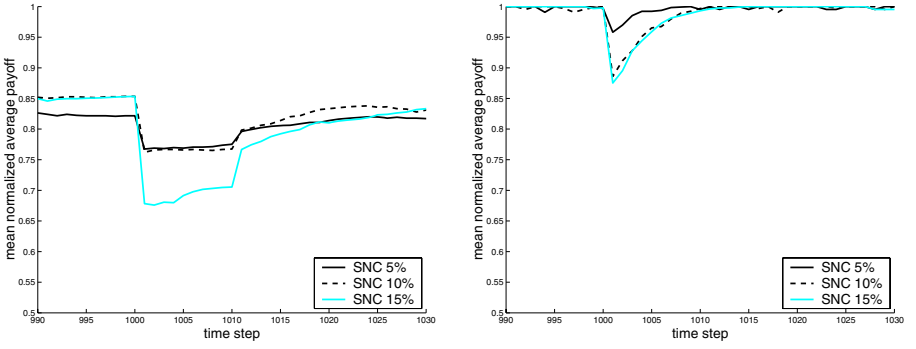


Fig. 5. Time evolution of the mean averaged payoff $\overline{\Pi}_D$ with different probabilites of cooperators becoming defectors (SNC) between time step 1001 and 1010; (a) synchronous, (b) asynchronous; population size: 40, $\sigma = 1$. Note the x and y axes scales.

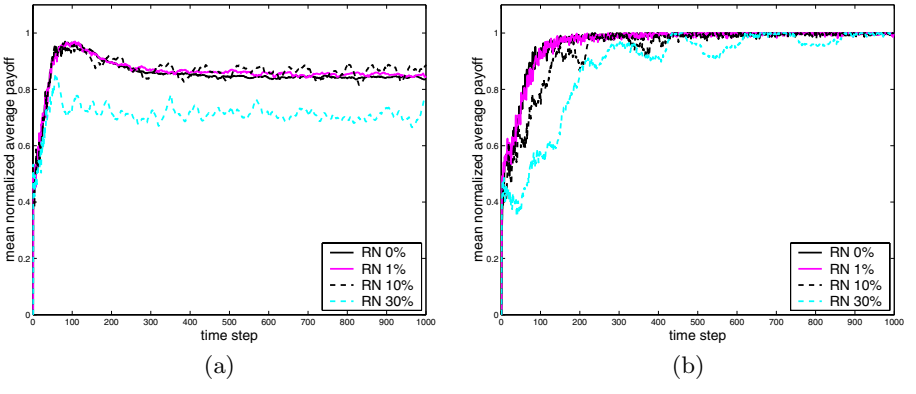


Fig. 6. Time evolution of the mean averaged payoff $\overline{\Pi}_D$ under different probabilities of rewiring errors; (a) synchronous, (b) asynchronous; population size: 40, $\sigma = 1$

prior to the removal of the perturbation. However, in the synchronous counterpart, the model never fully recovers from the perturbations. This is caused by the fact that the strategy error guarantees a rise of the number of D-D links in the same way shown on Fig. 3(a). Thus, in the synchronous case, all the runs will comprise a plummeting of $\overline{\Pi}_D$ in the noise interval if not earlier (see Fig. 4).

Rewiring Noise. Noise on the rewiring process consists for every player in making a mistake when rewiring and breaking off with a certain probability a link with a C-neighbor instead of a D-neighbor. This noise is introduced from the very beginning of the simulation.

The model has a good resistance to this type of noise as shown on Fig. 6. For both update mechanisms, high levels of noise (an error approximately 30

percent of the time) are necessary to notice important fluctations of $\overline{\Pi}_D$. Notice however, that the asynchronous model is once again more resistant to noise than the synchronous one. Indeed, the stochastic idiosyncrasy of the asynchronous updating process makes it more robust with respect to random errors.

5 Conclusions and Future Work

We have shown that, in spite of the fact that the players are not allowed to change their strategy, the network of relationships between players self-organizes towards a situation were the cooperators tend to cluster together and are surrounded by defectors, the phenomenon being more pronounced for high degrees of satisfaction. We also observe a greater overall stability of an asynchronous updating in comparison to the synchronous counterpart. Asynchronous updating equally shows higher resistance to two typical kinds of perturbations confirming its robustness. Future works will consists of letting the players change their strategy as well as modify their neighborhood.

References

1. R. Axelrod. *The Evolution of Cooperation.* Basic Books, Inc., New-York, 1984.
2. B. A. Huberman and N. S. Glance. Evolutionary games and computer simulations. *Proceedings of the National Academy of Sciences USA*, 90:7716–7718, August 1993.
3. K. Lindgren and M. G. Nordahl. Evolutionary dynamics of spatial games. *Physica D*, 75:292–309, 1994.
4. R. D. Luce and H. Raiffa. *Games and Decisions.* John Wiley and Sons, New York, 1957.
5. L. Luthi, M. Giacobini, and M. Tomassini. Synchronous and asynchronous network evolution in a population of stubborn prisoners. In G. Kendall and S. Lucas, editors, *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pages 225–232. IEEE Press, Piscataway, NJ, 2005.
6. M. A. Nowak, S. Bonhoeffer, and R. M. May. Spatial games and the maintenance of cooperation. *Proceedings of the National Academy of Sciences USA*, 91:4877–4881, May 1994.
7. M. A. Nowak and R. M. May. Evolutionary games and spatial chaos. *Nature*, 359:826–829, October 1992.
8. J. Maynard Smith. *Evolution and the Theory of Games.* Cambridge University Press, 1982.
9. M. G. Zimmermann, V. M. Eguíluz, and M. San Miguel. Coevolution of dynamical states and interactions in dynamic networks. *Physical Review E*, 69:065102(R), 2004.

Multi-agent-based Simulation for Formation of Institutions on Socially Constructed Facts

Takashi Hashimoto[1] and Susumu Egashira[2]

[1] School of Knowledge Science,
Japan Advanced Institute of Science and Technology (JAIST),
Nomi, Ishikawa, 923-1292, Japan
hash@jaist.ac.jp
http://www.jaist.ac.jp/~hash/index-e.html
[2] Otaru University of Commerce,
3-5-21, Midori, Otaru, 047-8501, Japan
susumue@ba2.so-net.ne.jp
http://www.res.otaru-uc.ac.jp/~egashira/

Abstract. In human societies, facts are constructed through social consensus. Here, the formation of social institutions in such a society is studied using a multi-agent-based simulation. Institutions are formed through communications among members, and the effects of errors in communication on the formation of institutions are investigated. Our results show that the institution is established when information suppliers frequently make errors in their information interpretation. We propose here that there is a phase transition in the error rate of the information suppliers in the formation of institutions.

1 Introduction

In the present study, we examine the formation of social institutions in a society using a multi-agent simulation. In particular, we investigate how errors in communication among members of the society affect the formation of institutions, when the "facts" emerge from interactions between the members.

One of the remarkable features of humans is that we live in societies and construct cultures. Here, we define culture as dominant modes of action and thought that are inherited through non-genetic mechanisms and are retained in a group of organisms. Among animals that form societies and cultures, human cultures are distinguished by their arbitrariness[1]. In animal societies, most cultures are related to survival and reproduction, such as methods of food utilisation and avoidance of enemies, while in human societies, a particular method or form is selected from among possible arbitrary options and regarded as formal, e.g., funeral rites and costumes used in rituals. This means that formality and correctness of codes of conduct and ethics in a society, such as morals and justice, are decided both unconsciously and unintentionally by members of that society.

Further, facts are sometimes determined by social consensus. For example, firms estimate their performance according to an accounting system. The measure of the estimation, such as the depreciation rate, is determined politically

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 675–684, 2005.

by agreement among public organisations. In a sense, the evaluation of a firm is largely dependent on the social consensus, and if a criterion is changed, the value of the firm is also changed without any accompanying physical change[2]. That is, facts about the firm's performance are constructed based on the social consensus. This character of facts is called "social construction of facts". It is discussed that our reality also depends on the social construction[3,4].

Systems that regulate our behaviour, such as the accounting system and laws, and that form the basis of our thoughts, such as customs and ethics, are called *social institutions*. Veblen[5] defined institutions as "settled habits of thought common to the generality of men". Social institutions are often made up through communication. People living in a complex society, in which little firsthand or direct information about various events is obtained, make their decisions according to information obtained from others. However, it is logically impossible to confirm the correctness of the information gained, because the confirmation of certain information requires additional information, which also requires additional information for confirmation, and so on. We called this character the "fundamental imperfection of information" and concluded that institutions work effectively to economise the cost for each person to confirm the correctness of information by believing the institutional systems established in the society[6].

Multi-agent-based simulations have been used to study the formation of institutions and norms[7,8,9,10]. These studies barely consider "fundamentally imperfect information" and the development of the agents' cognitive frameworks (world views or ways of thought) through interactions with others. Our previous work[11] showed that an institution as an ordered cognitive framework is formed as a result of social learning, such as the imitation of others' superficial actions and the continuous revisions of internal cognitive frameworks. However, in these studies, the social construction of facts and errors in the communication process are not considered. Thus, the present study was performed to investigate how institutions are formed when facts or partial facts are determined socially and how errors in information interpretation affect the formation of institutions.

Here, we suppose that people obtain information about some objective situations to be dealt with through communication, interpret it and act according to the interpretations. A typical example is the stock market where many investors make decisions about investments in stocks of various firms according to information supplied by securities companies and rating agencies. How their decisions are evaluated, i.e., their profit or loss, depends largely on the actions of all investors. Keynes[12] likened this situation to a "beauty contest" in which not only does the prize go to the person who receives the most votes but in which those who vote for the winner also benefit. Thus, it is thought that the validity of investors' decisions is partially socially constructed. To understand firms' performances, the investors must not only adequately select the information suppliers but also correctively interpret the information supplied. Note that the objective phenomena to be modelled in this paper are not limited to economic activities. As mentioned above, there are many activities in which codes

of conduct and ways of thinking are formed through communication and that affect our activities.

This paper is organised as follows: In Sec. 2, a multi-agent model is introduced. In Sec. 3, simulation results using the model are described. We discuss the results from the viewpoint of the formation of institutions in Sec. 4. The conclusions are presented in Sec. 5.

2 Model

We incorporated incorporate the social construction of facts and errors in information interpretation into our previous model[11]. The present model consists of two types of agents, information suppliers and information receivers, and objective situations with which the information receivers should deal. The information flows from the objective situations to the receivers through the suppliers, as illustrated schematically in Fig.1. An event sequence from setting an objective situation to evaluating the receivers' decision is called one turn.

Each agent has its own cognitive framework for interpreting information. The framework is expressed by a bit string $\boldsymbol{f}^S = (f_1^S, f_2^S, \cdots, f_L^S)$ for a supplier and $\boldsymbol{f}^R = (f_1^R, f_2^R, \cdots, f_L^R)$ for a receiver. Only the information receivers are aligned on a 2-dimensional $W \times W$ cell-plane with a periodic boundary.

The information suppliers observe the objective situations. One objective situation consists of L figures, each figure has two states: 0 or 1. The objective situation is expressed by a vector $\boldsymbol{O} = (O_1, O_2, \cdots, O_L)$. Each figure corresponds to each element of the frameworks, $\boldsymbol{f}^S$'s and $\boldsymbol{f}^R$'s. The objective situation is randomly generated at the beginning of each turn.

The information suppliers interpret the objective situations. The way of interpretation is implemented by the exclusive or (XOR) bit operation, defined as

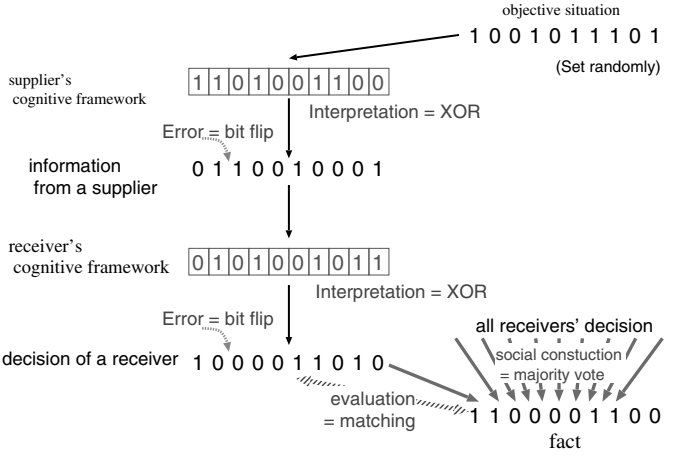


Fig. 1. The Information flow from an object to a supplier and to a receiver

$$XOR(x,y) = \begin{cases} 0\ (x=y) \\ 1\ (x \neq y) \end{cases} . \tag{1}$$

Errors in the interpretation are implemented by the bit flip operation, defined as

$$Flip(x) = \begin{cases} 0\ (x=1) \\ 1\ (x=0) \end{cases} . \tag{2}$$

The error function, $Flip$, is operated randomly on each bit according to the suppliers' error rate ε^S per bit. Thus, a supplier's interpretation, $\boldsymbol{I}^S = (I_1^S, I_2^S, \cdots, I_L^S)$, of information about an objective situation is expressed as:

$$I_i^S = Flip(\, XOR(O_i, f_i^S)\,) \ , \quad (i = 1 \sim L) \ . \tag{3}$$

Each information receiver, located on a 2D plane of size $W \times W$, adopts a supplier as the source of information about the objective situations. A receiver obtains information, $\boldsymbol{I}^S$, from the adopted supplier and makes its own interpretation, $\boldsymbol{I}^R$, using its cognitive framework, $\boldsymbol{f}^R$, in the same manner as the information supplier:

$$I_i^R = Flip(\, XOR(I_i^S, f_i^R)\,) \ , \quad (i = 1 \sim L) \ . \tag{4}$$

The error rate of the receivers is denoted by ε^R. The receivers make decisions based on their interpretation. In this paper, for simplicity, the decision is identified with the interpretation.

The receivers' decisions are evaluated in terms of facts, denoted by $\boldsymbol{F} = (F_1, F_2, \cdots, F_L)$, which are socially constructed through the majority vote,

$$F_i = Majority(I_i^R) = \begin{cases} 1 & \left(\sum_{\text{all receivers}} I_i^R > W^2/2\right) \\ 0 & (\text{otherwise}) \end{cases} . \tag{5}$$

That is, the fact for ith bit is 1 if more than half of the receivers interpret it as 1, and vice versa. Each receiver scores the number of bits in its own interpretation, $\boldsymbol{I}^R$, that match the fact, $\boldsymbol{F}$. Thus, the score of a receiver, denoted by P, is:

$$P = \sum_{i=1}^{L} (1 - XOR(I_i^R, F_i)) \tag{6}$$

After evaluation, each receiver compares its score with those of the eight neighbouring receivers. If a receiver has the lowest score alone, then a randomly selected element in its cognitive framework is altered; otherwise nothing happens. The lowest scored agent also changes its information supplier, adopting the supplier adopted by the best receiver among its neighbours. If more than one receiver has the best score in its neighbours, one is selected at random. That is, the locally worst receiver imitates the selection of supplier – i.e., the externally observable behaviour – of the locally best receiver and internally searches a better framework in a trial-and-error manner, which is the least smart and memory-less learning method.

3 Simulation Results

We report the results of simulation using the model described above. The conditions for the simulation were as follows. The size of the information receivers' plane was $W = 21$, and the numbers of receivers and of information suppliers were both 441. The length of the bit strings for the objective situations, the frameworks and the fact was $L = 10$. The initial cognitive frameworks were set randomly. While the receivers revised their frameworks according to the score, P, the suppliers did not change from their initial framework. Varying the error rates, ε^S and ε^R, we assess the effect of errors on the formation of institutions.

3.1 Erroneous Interpretation by Suppliers

To assess the effects of the suppliers' interpretation error on the formation of institutions, we conduct experiments in which only the suppliers make errors. The error rates are $\varepsilon^S = 1E-5, 1E-4, 0.001, 0.01, 0.1$ and 0; and $\varepsilon^R = 0$.

We observed how the receivers' selection of suppliers changed over time. A group in which the receivers adopted the same supplier is called a cluster. The size of the kth cluster, denoted by C_k, is the number of receivers in the cluster. The graph shown in Fig. 2(a) shows the dynamics of the size of the largest cluster, C, for various values of ε^S. Only in the case of $\varepsilon^S = 0.1$, the cluster size expanded rapidly and reached the maximum, $C_{max} = W^2 = 441$. Clusters hardly developed for other values of the error rate.

The degree of (dis)accordance between the cognitive frameworks of the receivers is measured by the Hamming distance between two receivers' frameworks

$$dist(r, r') = \sum_{i=1}^{L} \left| f_i^r - f_i^r \right| \ , \tag{7}$$

where r and r' represent two receivers and $|\cdot|$ is the absolute value. The average distance in the kth cluster,

$$D_k = \frac{1}{2C_k} \sum dist(r, r') \ , \tag{8}$$

is a measure of the (dis)accord of the framework in the cluster, where the sum is taken over all receivers, r and r', in the kth clusters.

Figure 2(b) shows the dynamics of the average distance in all clusters,

$$D_{ave} = \frac{1}{W^2} \sum_k C_k D_k. \tag{9}$$

Except in the case of $\varepsilon^S = 0.1$, the average distance converges to a value that depends in a straightforward manner on the error rate. Specifically, larger error rate were associated with grater distance. In the case of $\varepsilon^S = 0.1$, following the rapid growth of the largest cluster (around 7,000th turn), the receivers' frameworks begin to show accordance (around 8,000th turn) and finally become completely common to all the receivers ($D_{ave} = 0$, around 30,000th turn).

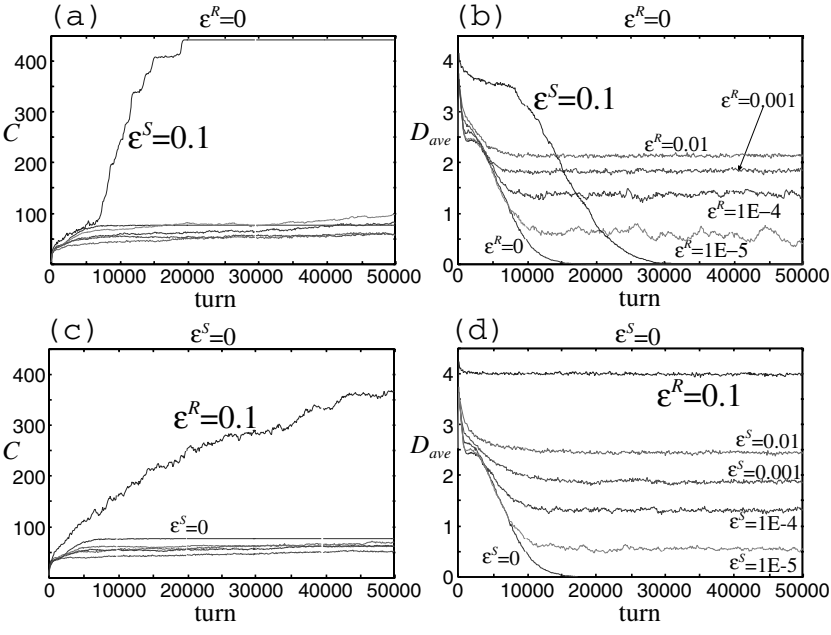


Fig. 2. The dynamics of the largest cluster size, C ((a) and (c)), and the average distance in clusters, D_{ave} ((b) and (d)), when the suppliers made errors in their interpretation process ((a) and (b)), and when the receivers made such errors ((c) and (d)). The x-axis is the turn. The error rates are $\varepsilon^S = 1E-5, 1E-4, 0.001, 0.01, 0.1$, and 0; and $\varepsilon^R = 0$ in (a) and (b); $\varepsilon^R = 1E-5, 1E-4, 0.001, 0.01, 0.1$, and 0; and $\varepsilon^S = 0$ in (c) and (d). All results are ensemble averages over 10 runs for each point.

3.2 Erroneous Interpretation by Receivers

The situation differs from the previous case when only the receivers make errors ($\varepsilon^R = 1E-5, 1E-4, 0.001, 0.01, 0.1$, and 0; and $\varepsilon^S = 0$). The dynamics of the largest cluster size, depicted in Fig. 2(c), is similar to the case of supplier's error, Fig. 2(a). Only when $\varepsilon^R = 0.1$, the largest error rate in the present experiments, a cluster expanded. However, the speed of growth of the cluster was much slower than in the case of suppliers' error. The average distance did not decrease at all for $\varepsilon^R = 0.1$, as shown in Fig. 2(d).

3.3 Social and Physical Facts

In our society, not all facts are constructed fully socially, but some are determined physically or externally. How a fact is determined socially or physically is a matter of gradient. Therefore, in addition to social construction, we incorporated the physical determination of facts into our model by identifying the facts with the objective situation. That is, if the ith bit of a fact is determined physically, then $F_i = O_i$. The degree of social construction is parameterised by $Sc = L_s/L$,

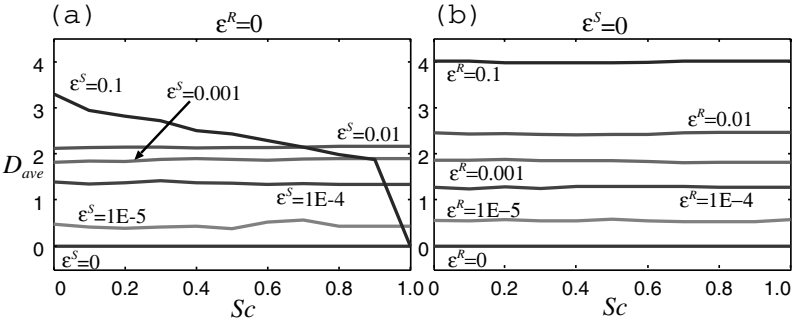


Fig. 3. Effects of the mixture of social and physical facts on (dis)accordance of cognitive frameworks. The x-axis shows the ratio of social construction in facts (Sc), and the facts are fully socially constructed at the right edge, Sc=1.0. The y-axis shows the average distance in clusters. (a) The suppliers make errors, $\varepsilon^R = 0$. (b) The receivers make errors, $\varepsilon^S = 0$. All results are ensemble averages over 10 runs for each point.

where $L_s$is the number of bits constructed by the majority vote, (5). In this study, each bit was fixed to either a social or physical fact according to the parameter Sc.

Figure 3 shows the average distances in clusters at the stationary states for various values of Sc for both cases of suppliers' (Fig. 3(a)) and receivers' errors (Fig. 3(b)). In the case of $\varepsilon^S = 0.1, \varepsilon^R = 0$ and $Sc = 1.0$, the receivers' frameworks come to complete accordance, as described in the previous section. This accordance is broken when no more than one bit is determined physically, i.e., $Sc < 1.0$. The mixture of social and physical facts affects on the formation of institutions only for $\varepsilon^S = 0.1, \varepsilon^R = 0$. The other lines in Fig. 3 are virtually flat for all values of Sc.

4 Discussion

4.1 Superficial and Cognitive Regularity

We showed that the effects of errors at an error rate of 0.1 on cluster formation and accordance of frameworks were different from the other error rates. Further, the effects also differed between suppliers' and the receivers' errors. The results for the social construction of facts are summarised in Table 1.

At an error rate of 0.1, the relative cluster size is 1 for both the suppliers' and the receivers' errors. This situation represents the formation of superficial regularity. Large-scale errors in the information supplied or of the receivers' interpretation cause large-scale fluctuations in the receivers' scores, which promotes the receivers' revision of suppliers. As the change is based on imitation of the externally observable behaviours of others, i.e., selection of the information supplier, the selections are finally canalised to one supplier.

Table 1. Effects of errors on cluster formation and the accordance of frameworks. The relative cluster size is the ratio of cluster size to all receivers.

error by	error rate	the relative cluster size	distance in clusters
suppliers'	< 0.1	0.1∼0.2	0.5∼2.5
	0.1	1	0
receivers'	< 0.1	0.1∼0.2	0.5∼2.5
	0.1	1	∼4.0

For suppliers' error, there is also regularity in the cognitive frameworks, $D_{ave} = 0$, when the error rate is $\varepsilon^S = 0.1$. Once all receivers obtain information from only one supplier, the interpretation error by the supplier does not matter, as the facts are constructed by the receivers themselves through a majority vote based on the uniform information in the society. Any receiver with a different framework from the majority must revise its framework to conform to the majority. Thus, the cognitive frameworks are built up until they are common to all receivers. This situation provides individuals in the society with consistency/regularity in their world views and has a self-enforcement function. Accordingly, we consider this situation the formation/establishment of a (cognitive) institution.

In contrast, frequent receiver error ($\varepsilon^R = 0.1$) does not result in concordance of the cognitive frameworks, i.e., $D_{ave} \cong 4.0$. The error prevents them achieving a good score and they keep changing their framework forever.

When the fact is partly determined physically, the formation of an institution cannot be completed. There remains diversity in the cognitive framework, $D_{ave} > 0$, as shown in Fig. 3. Suppliers' interpretation errors about the physically determined facts induce the receivers to revise their frameworks. With regard to physical facts, the revisions keep occurring as the receivers are not forced to come into accordance with the majority and frequent errors by the suppliers weaken enforcement upon receivers to conform to the same framework as the suppliers[1].

4.2 Phase Transition at Error Threshold

At an error rate of 0.1 per bit, the agents always make errors in interpretation as the frameworks are 10 bits in length. This value corresponds to the error threshold, $\varepsilon_{th} = 1/L$, which was proposed by Eigen[13] as the critical accuracy of information copy in the context of the origin of life. He showed that there was a phase transition at the critical value in the distribution of information entities[2].

Figure 4 shows thorough calculation of the dependence of the average distance, D_{ave}, on the suppliers' error rate, ε^S, and the degree of social construction,

[1] When there is no error, $\varepsilon^S = \varepsilon^R = 0$, the enforcement works well, as shown in our previous study[11], and indicated in Fig. 3 for all cases of the mixture of facts.

[2] The information entities are genes in the context of the origin of life.

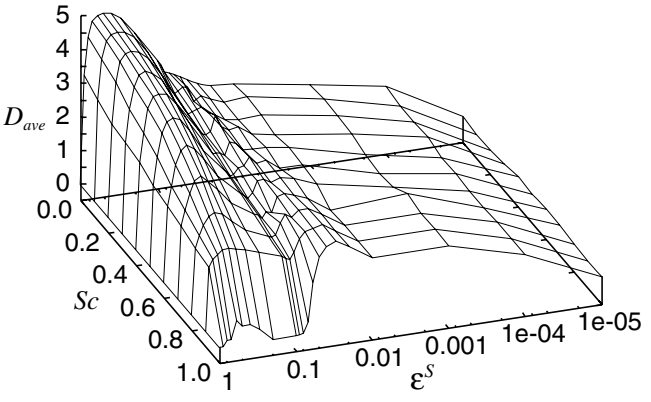


Fig. 4. Dependence of the average distance on the suppliers' error rate, ε^S (log scale), and the degree of social construction, Sc. These results are ensemble averages over 10 runs for each point.

Sc. The distance, D_{ave}, is an order parameter in this system, as complete accordance of the framework in clusters brings the distance $D_{ave} = 0$, and the random state $D_{ave} = L/2$. The distance increases monotonically with the error rate from small values to $\varepsilon^S \cong 0.03$. It decreases for large values of Sc at $\varepsilon^S = 0.03 \sim 0.1$. Especially, for $Sc = 1.0$, it falls abruptly to $D_{ave} = 0$, i.e., complete order. This abrupt descend indicates full accordance of the cognitive framework, i.e., social order, established at $\varepsilon^S = 0.1$. This result suggests a phase transition in the formation of institutions at the point of the error threshold of the suppliers' error, when facts are mainly constructed socially. In this graph, the change to $D_{ave} = 0$ is not completely discontinuous but is smooth. This may be caused by the finiteness of the system.

5 Conclusion

Using a multi-agent-based simulation, in which agents are equipped with adaptive cognitive frameworks, we studied the formation of institutions when the facts are constructed through social consensus.

Our simulation results suggested the following conditions for the formation of cognitive institutions, the ordered state of the cognitive framework in a society: 1) the information suppliers frequently make interpretation errors; 2) the information receivers seldom make interpretation errors; and 3) facts are constructed through social consensus. It was also suggested that there is a phase transition at the error threshold of the suppliers' error in the formation of institutions.

The present system remains to be improved in regard to several points. The model of agents is very simple and static, and we should therefore test how the results are reproduced with a more dynamic agent model, such as that of a cognitive individual with internal dynamics[14]. Another point is the introduction of a temporal correlation between objective situations and to let the

agents predict the future situation. This is also possible by using the individual model with internal dynamics. Future studies should also compare the suggested conditions to empirical evidence. Of course, while there are many difficulties in direct comparisons, political elections would be a possibility.

Acknowledgments. The authors thank Prof. Sawabe for valuable discussions and the anonymous reviewers for improving the paper. This work was supported in part by a Grant-in-Aid for Scientific Research (No.17651088) from the Ministry of Education, Culture, Sports, Science and Technology of Japan and by Japan Society for the Promotion of Science.

References

1. Merker, B.: From ape culture to ritual culture and language via vocal learning for song: The singular path to human uniqueness. S. Malloch and C. Trevarthen (Eds.), Communicative Musicality, Oxford University Press (forthcoming)
2. Sawabe, N: private communication
3. Berger, P.L., Luckmann, T.: The Social Construction of Reality: A Treatise in the Sociology of Knowledge, Anchor (1967)
4. Searle, J.R.: The Construction of Social Reality, Free Press (1995)
5. Veblen, T.B.: The Place of Science in Modern Civilisation and Other Essays, Huebsch (1919)
6. Egashira, H., Hashimoto, T.: The formation of common norms on the assumption of 'fundamentally' imperfect information. R. Conte, C. Dellarocas (Eds.), Social Order in Multiagent Systems, Kluwer (2001)
7. Conte R., Castelfranchi C.: Cognitive and Social Action, University College London Press (1995)
8. Epstein J.M., Axtell R.: Growing Artificial Societies, MIT Press (1996)
9. Conte R., Dellarocas C. (Eds.): Social Order in Multiagent Systems, Kluwer (2001)
10. Felix F., Polani D., Uthmann T., Modelling the emergence of possession norms using memes. Journal of Artificial Societies and Social Simulation **4**(4) (2001)
11. Hashimoto, T., Egashira, S.: Formation of social norms in communicating agents with cognitive frameworks. Journal of Systems Science and Complexity **14** (2001) 54–74
12. Keynes, J.M.: The General Theory of Employment, Interest, and Money, Macmillan Cambridge University Press (1936)
13. Eigen, M.: Selforganization of matter and evolution of biological macromolecules. Naturwissenschaften **58** (1971) 465-523
14. Sato, T., Hashimoto, T.: Dynamic social simulation with multi-agents having internal dynamics. K. Hashida, K. Nitta (Ed.) New Frontiers in Artificial Intelligence, Springer (forthcoming)

Extensions and Variations on Construction of Autoreplicators in Typogenetics

Kyubum Wee and Woosuk Lee

Ajou University, Suwon, S. Korea 443-749
{kbwee, sukky}@ajou.ac.kr

Abstract. Typogenetics was originally devised as a formal system with operations on DNA strands. It was recently demonstrated to be an effective model on which to study the emergence of self-replication by Kvasnicka et al.'s work. We make several extensions and variations on their work. The way of measuring difference and similarity between strands are improved. Many different mappings between doublet codes and their enzyme functions are tried. Triplet codes are also introduced. Through various experiments we observe frequent emergence of autoreplicators. We also find that emergence of self-replicators are robust phenomenon under various environments in typogenetics.

1 Introduction

Typogenetics is one of the approaches to studies on origins of life. It is a system consisting of strings originally devised by D. Hofstadter [3]. It was established as a formal system to study artificial life by Morris [8], and then it was demonstrated as a system where autoreplication can occur by Varetto [12, 13]. Kvasnicka et al. showed that construction of autoreplicators is a complicated combinatorial problem that must satisfy very involved constraints [4]. They pointed out that exhaustive enumeration is hopeless in discovering autoreplicators, and instead used evolutionary algorithms to construct autoreplicators.

We extended Kvasnicka et al.'s work in several ways. They used Hamming distance as a measure of similarity between two strands. We used minimum edit distance, and as a result we could construct many autoreplicators. We also tried various permutations of enzymes functions. We found that autoreplicators emerge regardless of particular associations between sub-strands and enzyme functions. Kvasnicka et al. used a doublet of nucleotides as a unit that encodes an enzyme. We tried triplets as units encoding enzymes. Autoreplicators still emerged, even though it took longer compared with the case of doublet encoding. All these results of our extension demonstrate that emergence of autoreplicators in typogenetics is quite a likely event under various environments.

2 Backgrounds

We briefly explain basic concepts of typogenetics and describe Kvasnicka et al.'s work. The alphabet of the typogenetics consists of four letters, called *bases*, A, C, G,

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 685 – 694, 2005.

and T. A and G are classified as *purines*, and C and T *pyrimidines*. A and T are *complementary*, and C and G are complementary. *Strands* are strings composed of four bases. A complementary strand S of a strand S has every base complementary to the corresponding base in S. For example, complementary strand of S = CCAGATTA is $\overline{S}$ = GGTCTAAT. A double strand, called DNA, consists of two strands S (lower strand) and R (upper strand) that are complementary, that is, $\overline{S} = R$. For example, D = $\begin{pmatrix} \text{CCAGATTA} \\ \text{GGTCTAAT} \end{pmatrix}$ is a double strand. A *quasistrand* is a strand with occasional hash symbols (#) which represent empty positions. For example, C##GATT# is a quasistrand. The distance between two quasistrands of the same length is the number of bases that are different from the corresponding bases divided by the length of a strand. More formally, the *distance* between two quasistrands $S = X_1 X_2 \cdots X_n$ and $R = Y_1 Y_2 \cdots Y_n$ is defined as $d = 1 - \frac{1}{n} \sum_{i=1}^{n} \delta(X_i, Y_i)$, where $\delta(X,Y) = \begin{cases} 1 & \text{if } X = Y \neq \# \\ 0 & \text{otherwise} \end{cases}$. For example, the distance between S = C##GATT# and R = C#TGACTG is $1 - \frac{1}{8}(1+0+0+1+1+0+1+0) = \frac{1}{2}$. It can be easily observed that $0 \le d(S,R) \le 1$ and that $d(S,R) = 0$ if and only if $S = R$.

Table 1. Mapping from doublet to *instruction* and *inclination*

no.	doublet	instr.	Inclin.	no.	doublet	instr.	inclin.
1	AA	mvr	l	9	GA	rpy	s
2	AC	mvl	s	10	GC	rpu	r
3	AG	mvr	s	11	GG	lpy	r
4	AT	mvl	r	12	GT	lpu	l
5	CA	mvr	s	13	TA	rpy	r
6	CC	mvl	s	14	TC	rpu	l
7	CG	cop	r	15	TG	lpy	l
8	CT	off	l	16	TT	lpu	l

Table 2. Functions of instructions

no.	instruction	function
1	*cop*	Enzyme turns on copy mode, until turned off, enzyme produces complementary bases
2	*off*	Enzyme turns off copy mode
3	*mvr*	Enzyme moves one base to the right
4	*mvl*	Enzyme moves one base to the left
5	*rpy*	Enzyme finds nearest pyrimidine to the right
6	*rpu*	Enzyme finds nearest purine to the right
7	*lpy*	Enzyme finds nearest pyrimidine to the left
8	*lpu*	Enzyme finds nearest purine to the left

A substrand of length two, called a *doublet*, is considered as a gene and expressed to be an *enzyme* performing a predefined function, called an *instruction*. Hence a strand can be translated into a stream of instructions. Table 1 shows the mapping from the doublets to the instructions. Table 2 describes the meaning of each instruction.

For example, the strand S = CCAGATTA is translated into the sequence of instructions mvl-mvr-mvl-rpy. It is denoted as instructions(S) = mvl-mvr-mvl-rpy.

The sequence of instructions corresponds to the *primary structure* of the enzyme. A *tertiary structure* is needed to determine the *binding site*. It is determined by the sequence of *inclinations* in Table 1. The inclinations l, s, and r represent left-turn, straight, and right-turn, respectively. The binding site is determined by the combination of the first inclination and the last arrow (Table 3). For example consider the inclinations sequence of the strand S = AGCGTTTG, which is s-r-l-l. The initial inclination (the zero-th inclination) is always assumed to be eastward. Hence the last arrow is upward (Figure 1).

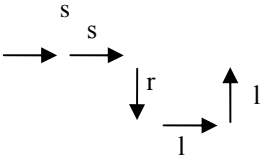


Fig. 1. An example of tertiary structure

Since the first inclination is s and the last arrow is upward, by Table 3, the binding site is the position of the first occurrence of C in whatever the strand to which the strand S is applied. It is denoted as binding-site(S) = C.

Table 3. Determination of binding sites

no.	1st inclin.	last arrow	binding	no.	1st inclin.	last arrow	binding
1	s	→	A	7	l	←	G
2	s	↑	C	8	l	↑	T
3	s	↓	G	9	r	↑	A
4	s	←	T	10	r	←	C
5	l	↓	A	11	r	→	G
6	l	→	C	12	r	↓	T

Now that the primary structure (the instructions) and the tertiary structure (the binding site) are defined, we denote the function of the strand S as enzyme(S) = (instructions(S), binding-site(S)). Now the replication process is defined as the process of applying the enzyme(S) to the strand S to produce the strand R, and denoted as replication(S) = R.

The following shows the replication process of the strand S = AGCGTTTG. Then instruction(S) = mvr-cop-lpu-lpy, binding-site(S) = s-r-l-l = (s, ↑) = C. Hence the replication process goes as follows:

$$\binom{\#\#\#\#\#\#\#\#}{AG\underline{C}GTTTG} \Rightarrow \text{mvr} \binom{\#\#\#\#\#\#\#\#}{AGC\underline{G}TTTG} \Rightarrow \text{cop} \binom{\#\#\#C\#\#\#\#}{AGC\underline{G}TTTG} \Rightarrow \text{lpu} \binom{\#CGC\#\#\#\#}{A\underline{G}CGTTTG} \Rightarrow \text{lpy}_{-} \binom{TCGC\#\#\#\#}{AGCGTTTG}.$$

A double strand $\binom{R}{S}$ is called an *autoreplicator* if replication(S) = R and replication(R) = S, in other words, the strand S is replicated to R and the strand R is replicated to S. The requirement that each of autoreplicator's complementary strands is replicated exactly into the other is a very restrictive constraint. Since there are 4^n different strands of length n, it is not feasible to search exhaustively for an autoreplicator. For n = 10, there are about one million different strands of length n. For n = 15, there are one billion strands, and for n = 20, one trillion strands, and so on.

An evolutionary method was used to construct autoreplicators. In order to evolve double strands to an autoreplicator, fitness of double strands is defined to measure how close the double strand is to an autoreplicator, and mutation operations are applied to the bases of strands. Fitness of a double strand $\binom{R}{S}$ is defined as follows:

fitness $\binom{R}{S} = \frac{1}{2}(2 - d(R, rep(S)) - d(S, rep(R)))$. It can be easily observed that $0 \leq$ fitness $\binom{R}{S} \leq 1$, and that fitness $\binom{R}{S} = 1$ if and only if $R = rep(S)$ and $S = rep(R)$, that is, $\binom{R}{S}$ is an autoreplicator. Recall that in a double strand $\binom{R}{S}$, $R = \overline{S}$. Hence the fitness of a strand S can be defined as follows:

$$\text{fitness}(S) = \frac{1}{2}(2 - d(\overline{S}, rep(S)) - d(S, rep(\overline{S}))).$$

Three kinds of mutation operators are introduced: *change*, *insertion*, and *deletion*. The following example illustrates these operations.

CCA$\underline{G}$ATTA → CCA$\underline{T}$ATTA	(change)
CCAGATTA → CCAGAT$\underline{C}$TA	(insertion)
CC$\underline{A}$GATTA → CCGATTA	(deletion).

The population size N = 1000, the lengths of the initial random strands ranged from 15 to 30, and the mutation rate continuously decreased from the initial mutation rate $p_{init} = 0.01$ to the final mutation rate $p_{final} = 0.001$ as the generation proceeds according to the rule :

$$p_{mut} = p_{init} - (p_{init} - p_{final}) \frac{current\ \ generation\ \ number}{number\ \ of\ \ total\ \ generations}.$$

The following is the autoreplicator thus obtained: $\binom{R}{S} = \binom{CGGCAGAAAAGAGT}{GCCGTCTTTTCTCA}$. First note that each of S and R is the complement of the other. The instruction sequence of S, according to Table 1, is rpu-cop-rpu-lpu-lpu-off-mvr. The inclination sequence is r-r-l-l-l-l-s. The first inclination is r, and the last arrow is downward $\downarrow$. Hence according to Table 3, the

binding site is the first occurrence of base T, which is at position 5 of the strand S. The following Figure 2 (a) shows the steps of the replication process of the strand S.

The instruction sequence of R is cop-rpu-mvr-mvr-mvr-rpy-lpu. The inclination sequence is r-r-s-l-l-s-l. The last arrow is ←. Hence the binding site is C. Figure 2 (b) shows the replication process. Figure 2 clearly demonstrates that $\binom{R}{S}=\binom{\text{CGGCAGAAAAGAGT}}{\text{GCCGTCTTTTCTCA}}$ is an autoreplicator.

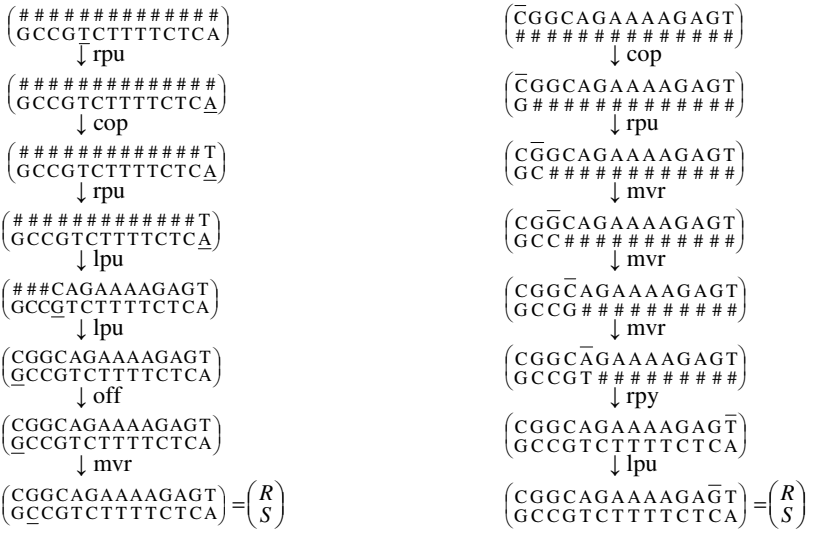


Fig. 2. (a) Replication of strand S. (b) Replication of strand R.

3 Improvements and Extensions

The previous section described Kvasnicka et al.'s work. We made three kinds of improvements and extensions on their work: refinement of the notion of distance between strands, shuffling of the functions of enzymes, and introduction of triplet codes for enzymes.

3.1 Refinement of Distance

We refined the definition of distance between two strands. Kvasnicka's definition of distance just compares the bases at the same positions of two strands, the bases at the first positions, the ones at the second positions, and so on. And if two strands are not of the same length, then the extra positions of the longer one are simply ignored. We employed minimum edit distance, or Levenshtein distance, as the measure of difference between two strands [l]. Minimum edit distance is also the way of comparing nucleotide sequences or amino acid sequences in bioinformatics [9, 10]. More formally, we defined the distance between two strands $S = X_1X_2\cdots X_n$ and

$R = Y_1 Y_2 \cdots Y_n$ as follows: $d = 1 - \frac{2}{m+n} lcs(S,R)$ where $lcs(S,R)$ stands for the length of the *longest common subsequence* of S and R. It is easy to see that $0 \le d \le 1$.

For example, let S = TGGACT and R = CGTGAT. Then the longest subsequence of S and R is GGAT, since GGAT is a subsequence of S and also a subsequence of R and there is not any common subsequence longer than GGAT. Hence the distance $d = 1 - 2\frac{4}{6+6} = \frac{1}{3}$. Recall that the Kvasnicka's distance is $d = 1 - \frac{1}{p}\sum_{n=1}^{p} \delta(X_i, Y_i)$, where $p = \min\{m,n\}$. The distance between S and R by the Kvasnicka's definition is $d = 1 - \frac{2}{6} = \frac{2}{3}$, since S and R coincide only at the second position and the sixth position. We believe that our definition reflects the similarity and difference between two strands better than Kvasnicka's. There is a simple and efficient dynamic programming algorithm that finds the longest common subsequence in $\theta(m+n)$ time [1].

3.2 Enzyme Functions

Kvasnicka's work found an autoreplicator under the fixed assignment of functions to enzymes. Table 1 in Section 1 shows the functions of enzymes. We tried many different associations of enzymes and their functions to see whether particular associations affect the chance of emergence of autoreplicators. We randomly permuted the instructions in the "instructions" column of Table 1. We found that emergence of autoreplicators were not affected by particular associations between enzymes and functions. The details are described in Section 4.

3.3 Triplets

Kvasnicka's work used doublets to encode enzymes. We experimented with triplet codes for enzymes to examine what difference they would make to the chance of emergence of autoreplicators and the lengths of autoreplicators. We found that it took more time for an autoreplicator to emerge, but the lengths of autoreplicators were not much longer than those with doublet enzymes. The details are described in Section 4.

4 Experiments

We implemented our algorithms on Pentium4 2.8GHz cpu with 1GB memory in Visual C++ language running on Windows XP Professional Service Pack1. For our evolutionary algorithm, we used the same mutation rates as the Kvasnicka's described in Section 2. The population size was set to 1000. The lengths of the strands in the initial population range from 15 to 30. They range from 15 to 45 for triplet codes. Note that length of a strand may be increased or decreased when mutated by the result of insertion or deletion operation. It may also stay still.

First, we compared the results obtained by using Hamming distance and edit distance. For doublet codes we used Table 1, and for triplet codes Table 4.

We ran the evolutionary algorithm for 100 times with doublet codes in Table 1. When the Hamming distance was used for distance between two strands, 8 autoreplicators emerged out of 100 runs. With the edit distance, 9 autoreplicators emerged out of 100 runs. We also ran the algorithm with triplet codes described in Table 4. With the Hamming distance 11 autoreplicators emerged out of 100 runs. With the edit distance 17 emerged out of 100 runs. As expected, we found that the edit distance was more effective than Hamming distance in measuring the difference between strands.

Table 4. Mapping from triplet to *instruction* and *inclination*

no.	triplet	instruct.	inclin.	no.	triplet.	instruct.	inclin.	no.	triplet.	instruct.	inclin.
1	AAA	mvr	s	23	CCG	rpu	l	45	GTA	mvl	s
2	AAC	lpy	s	24	CCT	lpu	l	46	GTC	lpy	l
3	AAG	lpy	r	25	CGA	lpy	l	47	GTG	cop	s
4	AAT	rpy	s	26	CGC	mvl	r	48	GTT	mvr	r
5	ACA	off	l	27	CGG	rpy	s	49	TAA	mvl	l
6	ACC	lpy	l	28	CGT	off	l	50	TAC	lpy	r
7	ACG	mvr	r	29	CTA	lpu	s	51	TAG	cop	r
8	ACT	off	r	30	CTC	rpy	r	52	TAT	mvr	s
9	AGA	mvr	s	31	CTG	rpy	l	53	TCA	off	l
10	AGC	lpy	l	32	CTT	lpu	l	54	TCC	rpu	l
11	AGG	rpy	r	33	GAA	rpy	r	55	TCG	rpu	s
12	AGT	rpu	l	34	GAC	lpu	r	56	TCT	mvl	l
13	ATA	mvr	s	35	GAG	mvl	s	57	TGA	cop	r
14	ATC	cop	r	36	GAT	mvl	s	58	TGC	rpy	l
15	ATG	mvr	r	37	GCA	mvr	s	59	TGG	mvr	s
16	ATT	lpu	s	38	GCC	rpu	l	60	TGT	rpu	l
17	CAA	mvl	r	39	GCG	rpu	r	61	TTA	mvr	r
18	CAC	mvr	s	40	GCT	mvr	l	62	TTC	mvl	s
19	CAG	mvl	r	41	GGA	rpy	l	63	TTG	mvl	l
20	CAT	lpy	s	42	GGC	lpu	s	64	TTT	rpu	r
21	CCA	lpu	l	43	GGG	mvl	l				
22	CCC	mvl	r	44	GGT	lpu	l				

For the rest of the experiment we used the edit distance as the measure of difference between strands. We ran the algorithm for 1000 times with the doublet instructions defined in Table 1. Autoreplicators emerged 86 times. We also ran it 1000 times with the doublet instructions randomly shuffled at every run. In this case, autoreplicators emerged 32 times. The same experiment was performed with triplet instructions. With the triplet instructions defined in Table 4, 31 autoreplicators were observed out of 1000 runs. With the instructions randomly shuffled at every run, 5 autoreplicators emerged.

Let us trace the autoreplication process of $\binom{R}{S}=\binom{\text{CGAAGAGCCTCCTCCA}}{\text{GCTTCTCGGAGGAGGT}}$ which is one of the 86 autoreplicators that emerged in our experiment with the doublet codes defined in Table 1. The instruction sequence of R is cop-mvr-rpy-rpu-off-mvl-rpu-mvr. The inclination sequence is r-l-s-r-l-s-l-s. The last arrow is $\leftarrow$. Hence the binding site is C.

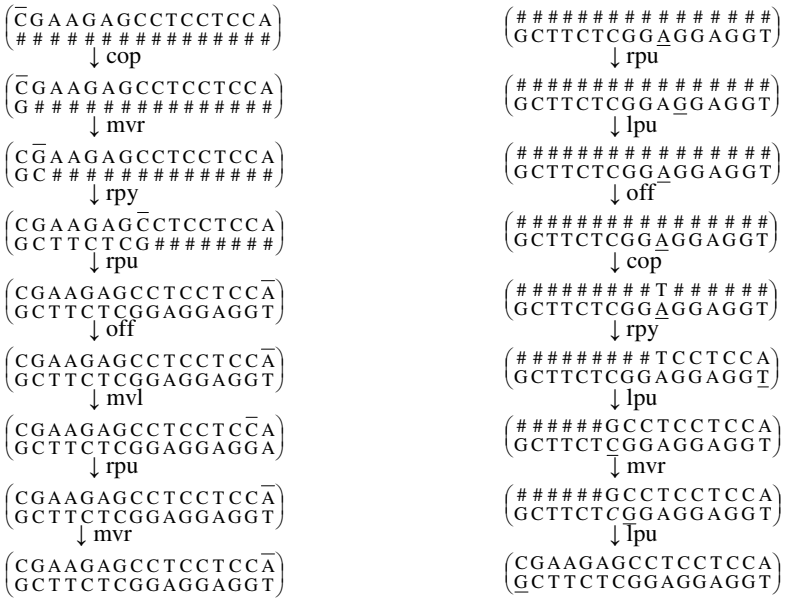


Fig. 3. Autoreplication process with doublet codes

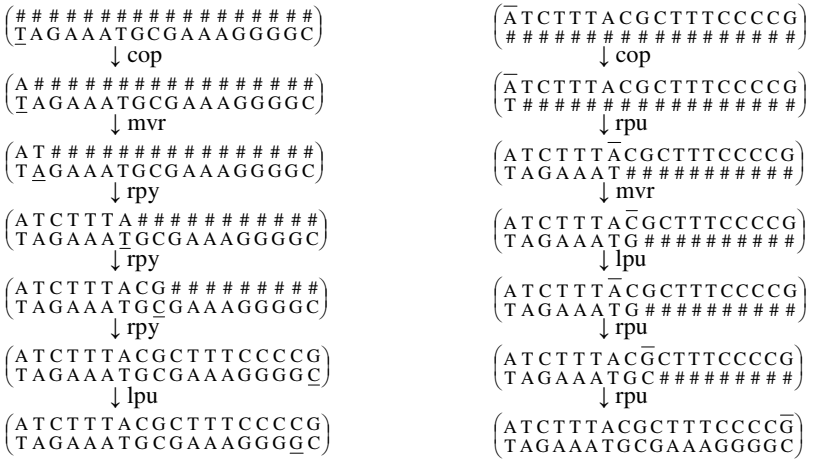


Fig. 4. Autoreplication process with triplet codes

The instruction sequence of S is rpu-lpu-off-cop-rpy-lpy-mvr-lpu. The inclination sequence is r-l-l-r-s-r-s-l. The last arrow is ↑. Hence the binding site is A. Figure 3 shows the autoreplication process with doublet codes.

Now let us trace the autoreplication process of $\binom{R}{S} = \binom{\text{A T C T T T A C G C T T T C C C C G}}{\text{T A G A A A T G C G A A A G G G G C}}$ which is one of the 31 autoreplicators that emerged in our experiment with the triplet codes defined in Table 4. The instruction sequence of S is cop-mvr-rpy-rpy-rpy-lpu. The inclination sequence is r-s-l-r-r-s. The last arrow is ↓. Hence the binding site is T.

The instruction sequence of R is cop-rpu-mvr-lpu-rpu-rpu. The inclination sequence is r-r-r-l-l-l. The last arrow is ↑. Hence the binding site is A. Figure 4 shows the autoreplication process with triplet codes.

5 Conclusions

Studies on self-replication have been carried out over the last fifty years. The models on which self-replication are studied include cellular automata, computer programs, strings, molecules, and even mechanical devices. Sipper provides an excellent survey on self-replication [11]. Typogenetics is one of the models based on strings. It has been around for over twenty five years, and recently its usefulness was rekindled by Kvasnicka's et al.'s work. We made several improvements and extensions on their work. Through various experiments we showed that spontaneous emergence of self-replicators is a robust phenomenon under varied environments.

We improved the way to measure the distance and similarity between strands. As a result, we could observe more frequent emergence of self-replicators. We tried many different mappings from the doublet enzymes to their functions. It was found that emergence of self-replicators were not affected by particular associations between doublet codes and their functions. We also introduced triplet codes instead of doublet codes. Although it took a little longer, self-replicators still emerged under various associations between triplet codes and their enzyme functions.

In our work, the set of instructions for the triplet codes stayed the same as in the case for the doublet codes. Our future work will include enlargement of repertoire of instructions for the triplet codes. Kvasnicka et al.'s work included studies on emergence of hypercycles [2]. We also plan to apply our improvements and extensions to the emergence of hypercycles.

Laing also studied self-replication on strings [6]. Sipper [2] noted, “Replication in Laing’s model is achieved by self-inspection, where the description of the object to be replicated is dynamically constructed concomitantly with its interpretation.” Our work evolves self-replicating strings under fixed rules. Morita and Imai used similar approach to Laing’s for constructing self-reproducing cellular automata [7], where “the machine can encode its shape into a description by checking its body dynamically.” Another work on evolving self-reproducing cellular automata is by Lohn and Reggia [5]. They evolved CA rules using genetic algorithms on a space of randomly strewn structures. Our work uses mutation only as a means of evolving self-replicating strings. We consider that it is appropriate to use mutations only in evolving self-replicating molecules.

Acknowledgments

This work was supported by the Brain Korea 21 Project and by the grant No. RT104-03-05 from Regional Technology Innovation Program of MOCIE of Korea.

References

1. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to algorithms* (2nd ed.), MIT Press, 2001.
2. M. Eigen and P. Schuster, *The Hypercycles: A principle of natural self-organization*, Springer-Verlag, Berlin, 1979.
3. D. Hofstadter, *Gödel, Escher, Bach: An eternal golden braid*, Basic Books, New York, 1979.
4. V. Kvasnicka, J. Pospichal, and T, Kalab, *A study of replicators and hypercycles by typogenetics*, ECAL VIII, pp. 37-54, LNCS 2159, Springer, 2001.
5. J. Lohn and J. Reggia, *Automatic discovery of self-replicating structures in cellular automata*, IEEE Transactions on Evolutionary Computation 1(3), pp. 165-178, 1997.
6. R. Laing, *Automaton models of reproduction by self-inspection*, Journal of Theoretical Biology 66, pp. 437-456, 1977.
7. K. Morita and K. Imai, *A simple self-reproducing cellular automaton with shape-encoding mechanism*. In C. Langton and T. Shimohara (Eds.), ALIFE V, 1997.
8. H. C. Morris, *Typogenetics: A logic for artificial life*, In C. Langton (Ed.), ALIFE I, pp. 369-395, Addison-Wesley, 1989.
9. D. Mount, *Bioinformatics: Sequence and genome analysis,* Cold Spring Harbor Laboratory Press, 2001.
10. P. A. Pevzner, *Computational molecular biology: An algorithmic approach*, MIT Press, 2001.
11. M. Sipper, *Fifty years of research on self-replication: An overview*, Artificial Life 4(3), pp. 237-258, 1998.
12. L. Varetto, *Typogenetics: An artificial genetic system.* Journal of Theoretical Biology 160, pp. 185-205, 1993.
13. L. Varetto, *Studying artificial life with a molecular automaton*, Journal of Theoretical Biology 193, pp. 257-285, 1998.

The Good Symbiont

Chrisantha Fernando

Dept. of Informatics, Center for Computational Neuroscience and Robotics,
University of Sussex, Falmer, Brighton, UK, BN1 9RH
ctf20@sussex.ac.uk
http://www.chrisantha.com

Abstract. A self-reproducing cycle has the fundamental organization, $A + X \longrightarrow 2A$, and is autocatalytic, i.e. the products catalyze the formation of the products. The rate of increase of A is proportional to A, i.e. exponential. Asexual living entities often grow exponentially when resources are abundant, and decay exponentially when resources are scarce, according to autocatalytic kinetics. If two previously independently replicating autocatalytic entities can form a physical union that is still capable of autocatalysis but with a reduced decay rate, then the symbiosis can be viable in an environment in which resources have been depleted, even if the symbiont has a lower growth rate than either of its component particles. A good symbiont possesses the following features: i. low steric hindrance between components, ii. policing of defection or cheating by symbiont components. iii. low decay rate back to components. iv. absence of emergence of active sites susceptible to decay reactions. v. high rate of the final reproductive step. Failure to form stable symbiosis can result from deficits in any of these features, and is a problem central to the origin of both metabolism and template replication.

1 Recycling Autocatalytic Systems

For persistence of the biosphere, the net 'biosphere metabolism' must be recycling [22]. Cells are autocatalytic systems, i.e. fluid automata composed of coupled autocatalytic cycles, at a stoichiometric level, that underly enzymatic catalysis [5]. Therefore we must explain how recycling autocatalytic systems evolved. This paper develops the work of G.A.M King on symbiosis of autocatalysts [1][2][3], by isolating 5 properties that a successful chemical symbiotic event requires.

Autocatalytic cycles must not be confused with 'autocatalytic sets', or 'reflexively autocatalytic systems', capable of 'collective autocatalysis' [7] [8], nor with other models of higher-order replicators [10][9][11]. Autocatalytic cycles consist of stoichiometricly described entities, consisting of small intermediates. Chemically feasible autocatalytic cycles of chemicals exist, for example, the Formose cycle [12], the reductive citric acid cycle [14], and the reductive citric acid cycle [15], but only the formose cycle 'bioid' has been implemented [13]. No reflexively autocatalytic system is known to exist unless superimposed upon an autocatalytic cycle. Even recent models of Kaufmann type autocatalytic sets ignore the

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 695–704, 2005.

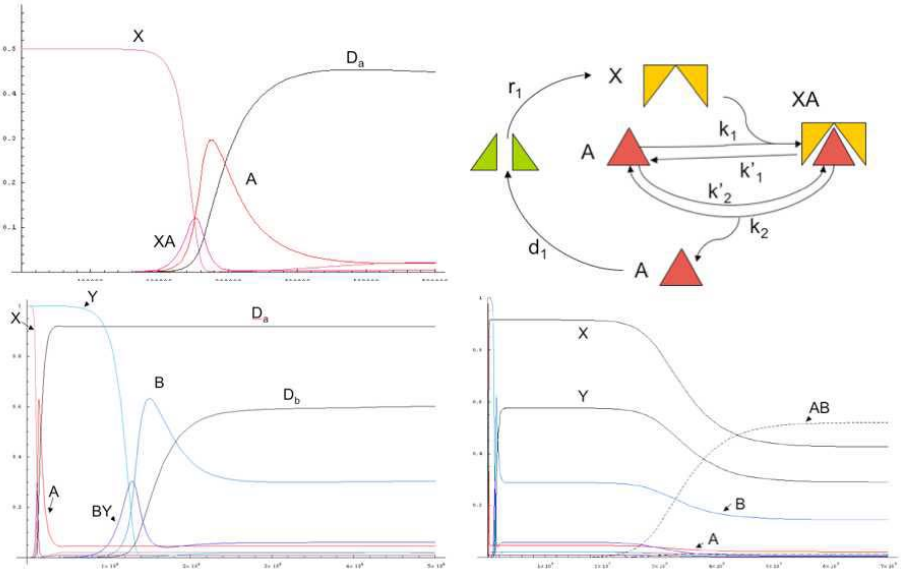


Fig. 1. Top Left: Concentration on the y-axis, and time on the x-axis. $k_1 = 0.001$,$k_1' = 10^{-8}$,$k_2 = 0.0001$,$k_2' = 10^{-7}$,$d = 2 \times 10^{-5}$,$r = 10^{-6}$. Initial $[X] = 0.5M$, Initial $[A] = 10^{-7}M$, and all other concentrations = 0. $[X]$ decays, being converted into A and XA, and eventually decaying to D, which is recycled to X. After an initial overgrowth of A, equilibrium is reached. **Top Right:** A simple recycling autocatalytic system. A uses the food molecule X forming the intermediate XA. A catalyses a conformational change in X to produce another A molecule. Therefore 2 A molecules pass out of the reaction. A degrades to D, and D is recycled by some environmental process, e.g using light energy, back into X. **Bottom Left:** Concentration on the y-axis, and time on the x-axis. For the autocatalytic particle B, the reaction network is of identical topology to particle A. $k_2 = 0.0001$, $k_2' = 10^{-8}$, $k_3 = 0.00001$, $k_3' = 10^{-8}$, $d_2 = 2 \times 10^{-6}$, $r_2 = 10^{-6}$. Initial $[Y] = 0.5$M, initial $[B] = 10^{-7}$ M. The concentrations of particle A and its products are included for comparison. Note particle A grows faster and earlier than particle B, but that particle B reaches a higher equilibrium concentration than A. **Bottom Right :** y-axis: Concentration, x-axis: Time. Note the long timescale compared to figures 2 and 3. A and B increase to reach their previous equilibrium concentrations. However, the particle AB (small dashed line) invades the population to eventually reach a higher concentration than A and B. YAB (large dashed line) exists at lower concentration, and $XYAB$ at even lower concentration (not visible).

problem of depleting side-reactions [16], which would turn otherwise neat autocatalytic sets into tar [17]. This problem must be faced head on when tackling autocatalytic cycles [18] [19]. 'Chemical' symbiosis has been explored recently by Fontana and Buss who describe a 'parasite route' to organization, where parasite 'autocatalytic cycles' incompletely deplete a core cycle. However, the likelihood of such a mechanism cannot be addressed with λ-calculus alone[20]. The aim here is not to contribute to the very sophisticated work on symbiosis [21], but

merely to confirm that symbiosis of autocatalytic cycles is what needs to be explained.

Consider the simple recycling system containing an autocatalytic replicator in the top right of figure 1. A is the growing state and XA is the reproducing state of the autocatalyst. This translates into the differential equations below.

$$X'(t) = -k_1 A(t)X(t) + k_1' XA(t) + r_1 DA(t) \quad (1)$$

$$A'(t) = -k_1 A(t)X(t) + k_1' XA(t) + 2k_2 XA(t) - 2k_2' A(t)^2 - d_1 A(t) \quad (2)$$

$$XA'(t) = k_1 A(t)X(t) - k_1' XA(t) - k_2 XA(t) + k_2' A(t)^2 \quad (3)$$

$$DA'(t) = d_1 A(t) - r_1 DA(t) \quad (4)$$

Figure 1 top left, shows that a stable equilibrium is reached in a recycling system. If the rate of recycling, r_1, is increased then the constituents of the cycle persist at a higher concentration. There is a fixed point at

$$DA = d_1 A / r_1 \quad (5)$$

$$X = (d_1 k_1' + d_1 k_2 + k_1' k_2' A)/k_1 k_2 \quad (6)$$

$$XA = (A(d_1 + k_2' A))/k_2 \quad (7)$$

When $[A] = 0$, $[X_{threshold}] = d_1 k_1'/k_1 k_2 + d_1/k_1$. This means that to maintain a non-zero concentration of A,$[X]$ must be greater than $[X_{threshold}]$. The threshold is increased as decay of A increases, and is decreased with increasing forward rates of reaction. Note that the rate of recycling of D into X, r, has no effect on $[X_{threshold}]$.

Figure 1 bottom left, shows another system with the same organization as above, but with a 10 times slower rate of uptake of reagent, Y, and a 10 times slower decay rate. Even though A grows more rapidly than B, at equilibrium a higher concentration of B is maintained at the same rate of recycling.

2 Symbiosis of Recycling Autocatalytic Particles

Now imagine that the two particles A and B can very rarely join together in the reaction $A + B \longrightarrow AB$, with a forward rate of as and a backward rate as' [1]. Assume that the system starts with a very small amount of AB (the symbiotic particle), i.e. 10^{-9}M, in the presence of a much larger amount of A (0.0001M), and B (0.0001M). For the moment let us prevent the formation of further AB by association between A and B, but still allow degradation of AB into A and

[1] King has elsewhere considered the symbiotic event to be the association not of A and B, but of XA and YB. Also he has compared the symbiotic state not with two previously independently replicating chemical species but with two species previously associated by mutual exchange of materials, i.e. the products of A are the food molecules of B and vice versa. The intermediate stage is not necessary (logically) for the emergence of physical union, although it may have promoted spatial associations between particles that would otherwise have not existed [3].

B at a low rate, $as = 10^{-7} sec^{-1}$. This simple autocatalytic system is shown in figure 2 and described by the equations below.

$$\begin{aligned}
X'(t) &= -k_1 A(t)X(t) + k_1' XA(t) + r_1 DA(t) + j_1' XYAB(t) \\
&\quad -j_1 YAB(t)X(t) && (8)\\
A'(t) &= -k_1 A(t)X(t) + k_1' XA(t) + 2k_2 XA(t) - 2k_2' A(t)^2 - d_1 A(t) \\
&\quad +as' AB(t) - asA(t)B(t) + j_2 XAB(t) + d_5 XYAB(t) && (9)\\
XA'(t) &= k_1 A(t)X(t) - k_1' XA(t) - k_2 XA(t) + k_2' A(t)^2 && (10)\\
DA'(t) &= d_1 A(t) - rDA(t) && (11)\\
Y'(t) &= -k_3 B(t)Y(t) + k_3' YB(t) + r_2 DB(t) - j_3 Y(t)AB(t) \\
&\quad +j_3' YAB(t) && (12)\\
B'(t) &= -k_3 B(t)Y(t) + k_3' YB(t) + 2k_4 YB(t) - 2k_4' B(t)^2 - d_2 B(t) \\
&\quad +as' AB(t) - asA(t)B(t) + d_3 YAB(t) + d_4 XYAB(t) && (13)\\
YB'(t) &= k_3 B(t)Y(t) - k_3' YB(t) - k_4 YB(t) + k_4' B(t)^2 && (14)\\
DB'(t) &= d_2 B(t) - r_2 DB(t) && (15)\\
AB'(t) &= asA(t)B(t) - as' AB(t) - j_3 Y(t)AB(t) + j_3' YAB(t) \\
&\quad -2j_0' AB(t)^2 + 2j_0 XYAB(t) + d_3 YAB(t) + j_2 XAB(t) && (16)\\
YAB'(t) &= j_3 Y(t)AB(t) - j_3' YAB(t) + j_1' XYAB(t) - d_3 YAB(t) \\
&\quad +d_5 XYAB(t) - j_1 YAB(t)X(t) && (17)\\
XYAB'(t) &= -j_0 XYAB(t) + j_0' AB(t)^2 - j_1' XYAB + j_1 YAB(t)X(t) \\
&\quad -d_4 XYAB(t) - d_5 XYAB(t) && (18)\\
XAB'(t) &= d_4 XYAB(t) - j_2 XAB(t) && (19)
\end{aligned}$$

If the reactions shown in Figure 2 can take place, AB will be autocatalytic. Imagine that in the particle AB, the subunit A_{ab} does not block the active site of the subunit B_{ab} for Y, and so let Y react with B_{ab} to produce YAB at the same rate $j_3 = k_3$, as B associates with Y to produce YB. Assume also that Y can dissociate from YAB at the same rate as Y dissociates from B (i.e. $j_3' = k_3'$). Assume that the subunit YB_{ab} on YAB has the same properties as YB, so that dissociation of YAB into $AB + B$ occurs at the same rate (i.e. $d_3 = k_4$) as dissociation of YA to produce $2B$. This is a tapping side-reaction of the autocatalytic cycle of AB. The reverse reaction $AB + B \longrightarrow YAB$ is ignored, as it can only help the replication of AB. Assuming that YB_{ab} does not interfere with A_{ab}s active site for X, then X can associate with YAB to form $XYAB$ at the same rate as A associates with X independently (i.e. $j_1 = k_1$). The same is true for the reverse reaction (i.e. $j_1' = k_1'$). Assume that $XYAB$ is subject to two possible side-reactions. Let it decay to $B + XAB$ at rate $d_4 = k_4$, and into $A + YAB$ at rate $d_5 = k_2$, again assuming no steric interference between the active sites of subunits XA_{ab} and YB_{ab}. XAB then reacts to form AB and A at rate $j_2 = k_2$. The formation of XAB by AB binding to X, before Y binds to

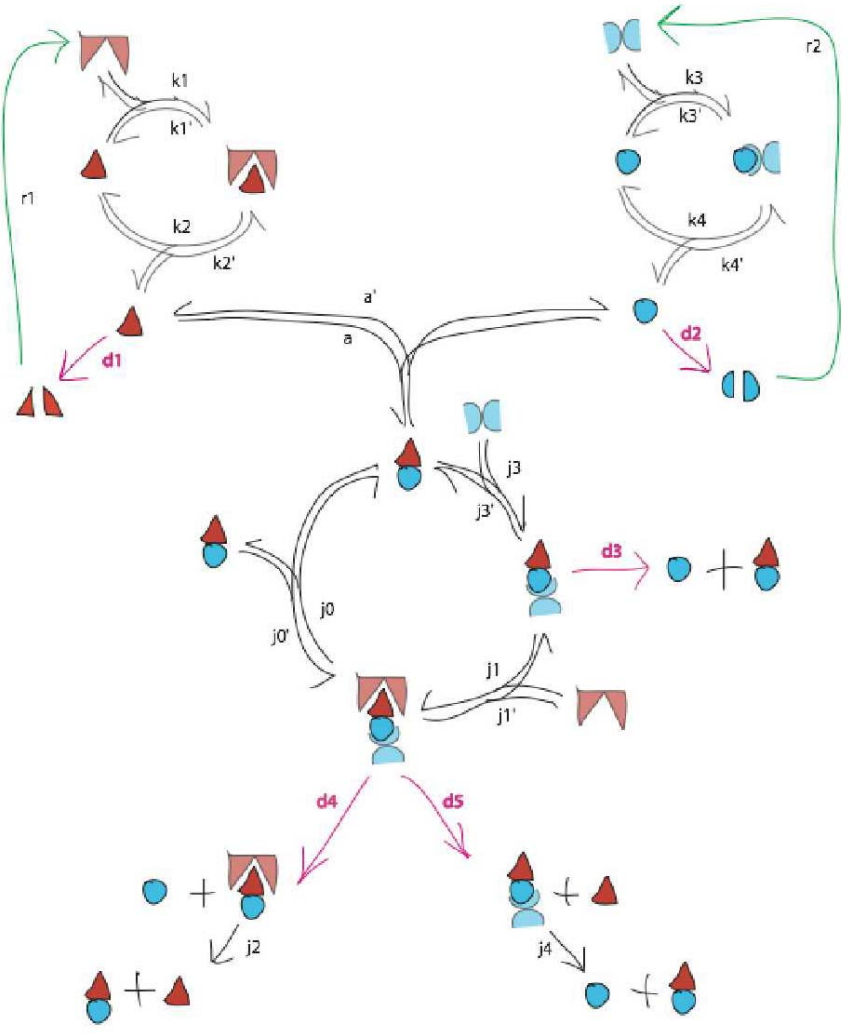


Fig. 2. Shows the higher order autocatalytic cycle of AB, the independent cycles of A and B, and the corresponding decay reactions. A is the red triangle, B is the blue circle. X is a pale red double triangle shape, and Y is the pale blue pair of hemi-circles. Complexes are shown as combinations of these elements. Rate constants correspond to those described by eqn. 8 to 19.

AB, has been ignored. This would either produce another possible autocatalytic cycle with X binding first and Y binding second, or produce a non-cycling decay chain. The consequences of this alternative cycle are significant but here it is assumed that XAB cannot re-enter the cycle apart from by further decay into food molecules. $XYAB$ can complete the autocatalytic cycle by promoting the simultaneous uni-molecular conformational change of $XYAB$ into $AB + AB$. It

is assumed that this occurs at the forward rate of $jz = 0.001 = k_2 > k_4$, although it may be more reasonable to expect that it would take the sum of the average period of dissociation of A from XA and B from YB, i.e. $(k_1^{-1} + k_2^{-1})^{-1}$).

Figure 1 bottom right, shows the concentrations of substrates for the reaction system described above . Each species undergoes exponential growth followed by depletion. Growth of the symbiont is delayed not because it must be produced by further symbiosis (these are forbidden), but because of its initial low concentration and lower growth rate than A or B. Its growth is limited by the X and Y concentrations that have been established previously by the growth of A and B. Neverthaless eventually AB outgrows A and B, and A and B continue to persist at low concentrations.

3 Kinetic Analysis

How do the kinetic properties of the particle AB affect the equilibrium concentration of AB? The system behaviour in figure 1 bottom right, is dependent on the following assumptions.

1. A_{ab} does not interfere with the binding of Y to B_{ab}, and B_{ab} does not interfere with the binding of X to A_{ab}. This means $j_3 = k_3$, $j_1 = k_1$ and $j_3' = k_3'$, $j_1' = k_1'$.

2. A_{ab} does not interfere with the production of B from YAB, nor from $XYAB$. B_{ab} does not interfere with the production of A from $XYAB$. This means that $d_3 = d_4 = k_4$ and $d_5 = j_2 = k_2$.

3. The rate of production of $2AB$ from $XYAB$ is equal to the rate of production of $2A$ from XA. This means $jz = k_2 > k_4$. jz' is negligible.

4. The decay of AB into A and B, (as), occurs at a low rate, 10^{-7}, and the rate of production of AB from A and B, (as), is zero. The results are similar if the rate of production, (as), of AB is 10^{-14} and the initial concentration of AB is 0.

5. AB does not undergo emergent decay reactions caused by novel active sites uncovered by their union. This means no alternative decay reactions of AB other than those above have been considered.

As with any autocatalytic cycle, the concentration of constituents reaches a fixed point when the rate of production of constituents equals the rate of decay of the constituents. The fixed point $[AB^*]$ is determined by the assumptions, which are now relaxed.

3.1 Interference Between Components

Assume that S_{ab} is the steric hindrance of A upon the binding of Y to B, and $S_{ba}$is the steric hindrance of B upon the binding of X to A. Both range from 0 to 1, 1 being maximal hindrance and 0 being no hindrance. Let $j_3 = (1 - S_{ab})k_3$ and $j_1 = (1 - S_{ba})k_1$. What happens to the equilibrium concentration of AB as S_{ab} and S_{ba} are varied? Figure 3 top left, shows the equilibrium concentration of AB for different values of S_{ab} and S_{ba}. At high values of steric hindrance $[AB^*]$ is virtually zero.

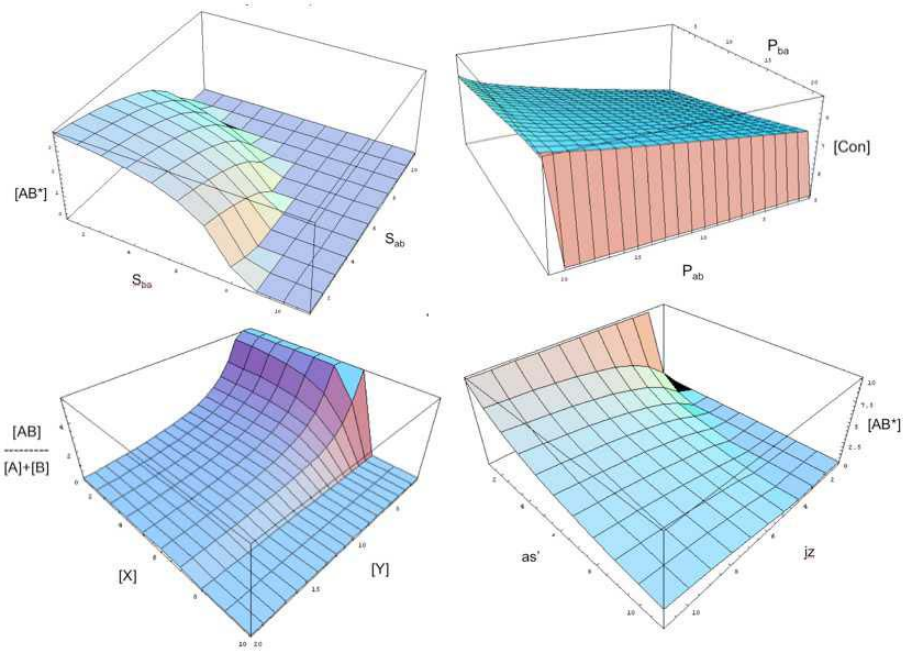


Fig. 3. Top Left: Steric Hindrance between components can prevent symbiont replication. $[AB^*]$ drops off rapidly at high rates of hindrance resulting in very low concentrations of AB. **Top Right:** The z-axis shows not [AB*] alone, but the summed equilibrium concentration of all configurations of AB, i.e. YAB, XYAB, AB, and XAB. The symbiont has greater concentration with greater amounts of protection of premature detachment. There is an anomaly at high P_{ba}, where $[AB^*]$ actually decreases rapidly. This is associated with an increase in $[Y]$ and $[XAB]$ and a decrease in $[YAB]$, $[AB]$ and $[XYAB]$ and is explained because the cycle is tapped by XAB which cannot be recycled with maximal policing of A by B. **Bottom Left:** The z-axis shows the ratio of AB particles to A and B particles. At low $[X]$ and low $[Y]$, AB dominates, but increasing either one makes the simpler particles more successful. The sensitivity of $[AB^*]$ to increasing $[X]$ is sharper than its sensitivity to increasing $[Y]$. **Bottom Right :** jz was increased from 0 to k_4 in 100 increments. There is a threshold of jz below which $[AB^*]$ is very low. This threshold depends on as'. as' was increased from 0 to 10^{-7} and as expected $[AS^*]$ decreased as decay of AB into A and B increased.

3.2 Policing by Components

Assume that A_{ab} is actually able to prevent the premature detachment of Y from YAB to produce B, and also of Y from $XYAB$, by an equal amount. Therefore let $d_3 = d_4 = k(1-P_{ab})$ where P_{ab} is the extent of protection or policing afforded by A_{ab} to Y. Also let B_{ab} protect the X particle from premature detachment from XAB and $XYAB$, so that $d_5 = j_2 = k_2(1 - P_{ba})$. Assume that S_{ab} and S_{ba} are reset to 0 as before.

As expected figure 3 top right, shows that $[AB^*]$ increases with increasing policing by A_{ab} of B_{ab} and B_{ab} of A_{ba}. However, at very high values of P_{ba} the cycle is drained into XAB which cannot decay. This anomaly is due to our assumption that XAB can only take part in one decay reaction. Note the build up of XAB by the irreversible reaction at rate d_4, but since $P_{ba} = 1$ then $j_2 = 0$ so XAB builds up.

3.3 Reversion to Components, and Reproduction Rate

Let us reduce the rate of the reproduction reaction producing $2AB$ from $XYAB$. Let $jz = Rk_2$ where R ranges from 0 to 1. Also let us make $as' = 10^{-7}F$, where F ranges from 0 to 1. Reset S_{ab}, S_{ba}, P_{ab} and P_{ba} to 0. Figure 3 bottom right, shows the results we expect, that decreasing as' increases $[AB^*]$, as does increasing jz. Note that below a threshold of jz, $[AB^*]$ decreases rapidly to zero.

3.4 The Effect of Resource Concentration

AB competes for the same reactants as A and B. All follow the overall growth law shown below, where C is the constituent concentration, β the growth rate, α the decay rate, and R the resource concentration, $dC/dt = -\alpha + \beta R$. When R is high, growth is dominated by β, but when R is low, decay (the α term) predominates. The growth of the fastest constituent (A in this case) depletes R, and starts the process of decay rate based selection, where particles with low α decay more slowly. This simple process is slightly complicated due to the presence of two resources X and Y. Figure 3 bottom left, shows the effect of increasing X from 0 to 0.03M and Y from 0 to 0.02M, and keeping them fixed at these values by magical topping up. Each particle is started at the same initial concentration, 0.0001M. The z-axis shows the ratio $[AB]/([A] + [B])$.It shows that AB obtains higher values relative to A and B, at low resource concentration.

4 Conclusion

If the symbiont can reduce its decay rate relative to the composing particles decay rates, then it can obtain higher equilibrium concentrations. However, at high resource concentration selection is on the basis of growth rate, not decay rate, and here A and B out-compete AB, since growth is exponential, and the growth rates of A and B are greater (under reasonable assumptions) than the growth rate of AB. Once AB depletes X and Y to even lower levels, another autocatalytic particle may arise which utilizes a wider range of food molecules, e.g. $ABCD$, using P and Q as food, if appropriate particle varieties exist. The probability of further symbiosis depends on particle structure. How can the specificity of reactions required for a successful symbiont be achieved? In chemistry this is done by addition reactions that consume active sites. King proposes that the coordination compounds of metal ions in aqueous solution have the appropriate properties.

The problem of reducing the decay rate of the autocatalytic particle is a very general one for the origin of life. It is encountered in the origin of metabolism in the problem of how the autocatalytic formose cycle can be viable (e.g. in the chemoton) when it is subject to so many side-reactions, and where no enzymes are yet available [4][5]. The side-reaction problem is surprisingly also faced in non-enzymatic template replication, in which a chain of coupled autocatalytic cycles of replicating templates are tapped by elongating side reactions. How the side-reaction problem can be solved defines a continuing research program [6].

Acknowledgments. Thanks to Eors Szathmary, Peter Ittzes, Szabolcs Szamado, Zoltan Szatmary and Katalin Csepi for their kindness.

References

1. King, G.A.M.: Autocatalysis. Chem. Soc. Rev. **7** 1978 297–316
2. King, G.A.M.: Recycling, Reproduction and Lifes Origins Biosystems. **15** 1982 89–97
3. King, G.A.M.: Symbiosis and the Origin of Life. Origins. Life **8** 1976 39-53
4. Ganti, T.: The Principles of Life. Oxford University Press 2003
5. Ganti, T.: Chemoton Theory Vol I and II. Kluwer 2004
6. Maynard-Smith, J. , Szathmary, E.: The Major Transitions in Evolution. Oxford University Press 1995
7. Kauffman, S. Autocatalytic Sets of Proteins. J. Theoret. Bio **119** 1986 1-24
8. Farmer,J.Doyne, StuartA.Kauffman, and Norman H. Packard Autocatalytic replication Polymers. Physica D **22** 1986 50-67
9. Stadler, B.M.R. and Stadler, P.F. Molecular Replicator Dynamics Adv. Complex Systems **6** 2003 47-77
10. Eigen M. and P. Schuster. The Hypercycle: A principle of natural self-organization. Springer, Berlin, 1979
11. Richard E. Michod Darwinian Dynamics: Evolutionary Transitions in Fitness and Individuality Princeton University Press. 1999
12. Breslow, R. Tetrahedron Letters /bf 21 1959 22-26
13. Decker, P. Schwoer, H. Pohlmann, R. Bioids Journal of Chromatography **224**1982 281-291.
14. Wachterschauser, G. Microbiol. Rev **52** 1988 452-484
15. Smith, E. Morowitz, H.J. Universality in Intermediary Metabolism. PNAS **101**36 2004 13168-13173
16. Mossel, E. and Steel, M. Random Biochemical Networks: the probability of self-sustaining autocatalysis. Journal of Theoretical Biology (forthcoming).
17. Taming Combinatorial Explosion. PNAS **97**14 2000 7678-7680
18. Szathmary, E. The Evolution of Replicators Philos. Trans. R. Soc. Lond B. Biol Sci. **29** 355 2000 1669-1676

19. Szathmary, E. Santos, M. and Fernando, C. Evolutionary Potential and Requirements for Minimal Protocells Topics in Chemistry (forthcoming).
20. Fontana, W. and Buss L.W. "The Arrival of the Fittest" : Toward a Theory of Biological Organization. Bulletin of Mathematical Biology **56** 1994 1-64.
21. Margulis, L. Symbiosis in Cell Evolution 2nd Edition. Freeman, New York. 1993.
22. Lovelock, J.E. Gaia: A New Look at Life on Earth Oxford University Press, 1979.

Self-description for Construction and Execution in Graph Rewriting Automata

Kohji Tomita[1], Satoshi Murata[2], Akiya Kamimura[1], and Haruhisa Kurokawa[1]

[1] National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan
{k.tomita, kamimura.a, kurokawa-h}@aist.go.jp
[2] Tokyo Institute of Technology, Yokohama, Japan
murata@dis.titech.ac.jp

Abstract. In this paper, we consider how self-description can be realized for construction and execution in a single framework of a variant of graph rewriting systems, called graph rewriting automata. As an example of self-description for construction, self-replication based on a self-description is shown. A meta-node structure is introduced for self-description for execution that enables us to embed rule sets in the graph structure. It can be a model of systems that maintain and modify themselves.

1 Introduction

Although precise definition of self-description is difficult, some systems both in nature and artificial world seem to utilize properties of self-description. One of them is apparently genome information for living things. Living things are constructed using genetic information by appropriate construction mechanism embedded in cells, and it can be viewed as a kind of self-description for construction. More specifically, the family of aminoacyl-tRNA synthetase can be considered as a constructor of proteins from RNA by associating codons to amino acids. Their blueprint is also described in DNA and interpreted by themselves. This information is transferred when cells are divided. Before the discovery of DNA structure [14], a similar mechanism to such genetic information was devised by von Neumann as a blueprint of self-reproducing cellular automata [13]. After that, much study has been done on self-replication in two dimensional cellular automata [9,10].

Other examples of self-description in computer science are reflective programming languages such as 3-Lisp [11]. Information on computation (environment and continuation in 3-Lisp) to execute programs is described explicitly in the same level, and can be referenced and altered by constructing reflective tower as much as needed. It is viewed as a self-description for execution. These self-descriptions have potential for increasing the stability and flexibility of the systems, but both are not realized simultaneously.

In this paper, we consider how self-description can be realized for construction and execution in a single framework of a graph rewriting system [3]. Graph rewriting systems or graph grammar studies rewriting of graphs according to

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 705–715, 2005.

some rewriting rules. It can be regarded as generalization of formal grammars on strings and cellular automata. Thanks to its expressibility for dynamic behavior of various structures, it has been studied in wide area. Recently, it is used for self-reconfigurable robots [5,7], artificial chemistry [2] and also self-replicating systems [12,4]. Usually rules of such systems are defined outside of the graph structures, and many examples are based on elaborate rule design. However, direct dependence on the rules outside makes it difficult to construct entities that have autonomy. Considering self-replication, processes without such independence are like crystal growth. Embedding the rules, at least in part, in the structure can introduce some kind of hierarchy and enhance its autonomy on the upper level. In order to realize self-replication process of graph structures using self-description, we define operations for graph rewriting, introduce a construction arm, and design a sequence of operations to construct itself encoded in the structure.[1] Along this design, we can envision evolvability of systems by introducing modification of the program codes [6].

In the following, after introducing the framework in Sect. 2, we show self-replicating process based on self-description for construction in Sect. 3. Then, we suggest that the self-description for execution can be realized in our framework in Sect. 4. Section 5 concludes the paper.

2 Graph Rewriting Automata

In this section, we show the framework we consider in this paper. It is a variant of graph rewriting systems, and called graph rewriting automata. We assume that the base graph structure is a 3-regular graph, i.e., each node has three neighbor nodes. Each node has an internal state chosen from a finite set. At each node, a cyclic order of links is defined. Graphs are rewritten by state transition, as in state transition in cellular automata, and by structure rewriting. We use five kinds of graph rewriting rules as shown in Fig. 1. Division rule divides a node into three nodes. Commutation rule rotates a pair of nodes counter-clockwise. Annihilation rule eliminates a pair. Swap rule changes the local connection relation. Note that application of rules except for swap rule preserves planarity of graphs. Also, there is asymmetry in swap rule, i.e., 'swap n_1 n_2' and 'swap n_2 n_1' produce different results — upper or lower links crossed in the figure.

An initial graph and a set of rules determine the development process. Rules are applied synchronously on the time of natural numbers; all the node rules (state transition and division) at even time and all possible link rules (commutation, annihilation and swap) at odd time.

[1] Uniform treatment of description and machine is addressed in Salzberg *et al.* [8]. They introduced tangled construction hierarchy, and examined graph-constructing graphs on the basis of directed graphs using a construction head. In this paper, we assume undirected 3-regular graphs as underlying structures and focus on self-replication process in detail. In particular, in our system, the construction head is also realized as a sub-graph, and it can be constructed in the course of graph rewriting.

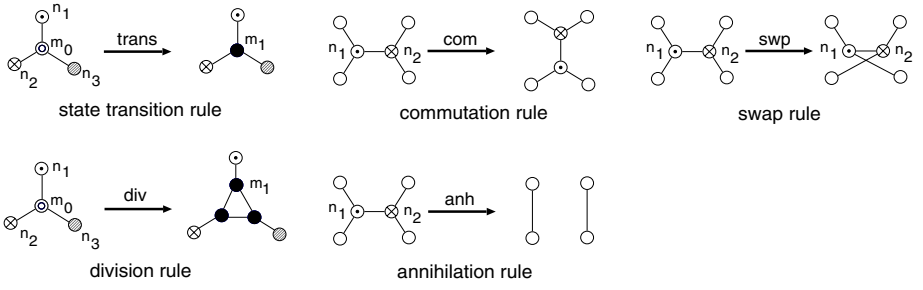


Fig. 1. Rules of graph rewriting automata; two node rules (trans and div) and three link rules (com, anh and swp)

3 Construction Using Self-description

3.1 Outline of Self-replication Process

In this section, we show how self-replication is performed based on its self-description for construction. The idea is to introduce a construction arm with three components: a manipulator, a controller and a reader. The manipulator performs modification on the graph structure, and the reader reads a coded plan (program) so that the manipulator can construct a structure according to the program sequentially. The controller controls their execution by switching its modes. Figure 2 illustrates the structure of the construction arm (left) and a schematic of the manipulator located in a usual state (middle). As in this figure, the manipulator usually indicates a target node. The base graph is undirected, but the directions of the manipulator and the reader are introduced using different states for the heads and tails of them. They are represented by the arrows in the figure. In the following, when we illustrate the motion of the manipulator, it is indicated just as an arrow (Fig. 2(right)). In this section, black circles indicate nodes of the construction arm, and white circles indicate other nodes. (This distinction is not fixed and, according to the internal states, the role changes.)

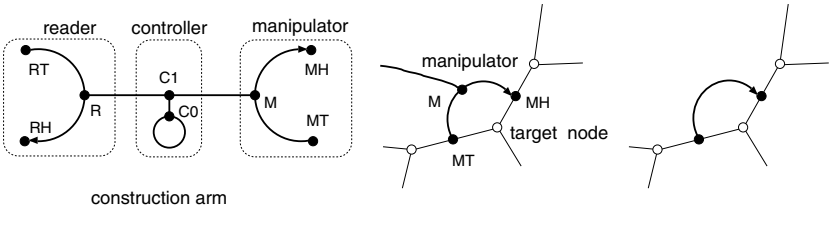


Fig. 2. Structure of construction arm

The manipulator executes the following eight operations on the target node.

– State transition operations: Reset, Inc

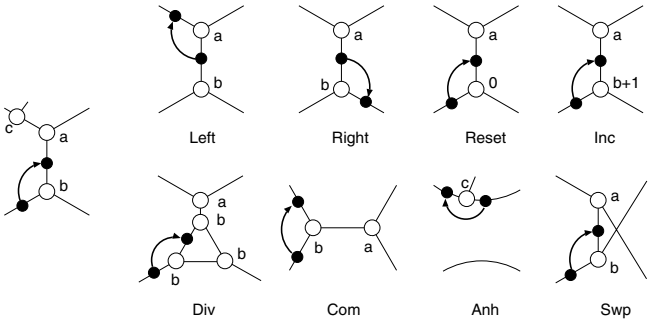


Fig. 3. Execution of eight operations

– Move operations: Left, Right
– Operations of executing graph rewriting rules: Div, Com, Anh, Swp

We assume that states of the nodes are chosen from $\{0, ..., N-1\}$, and Inc operation is for increment (modulo N). We distinguish these operations from the corresponding graph rewriting automata rules (div, com, anh, swp) by capitalizing. The effect of each operation is summarized in Fig. 3. In this figure, the node with state c is omitted except for Anh operation from the result.

Figure 4 illustrates the outline of the whole process of self-replication. It resembles the one proposed by von Neumann [13]. At the initial state, two components are connected by the construction arm: a circular ladder structure (or, simply, a ring) storing programs, and a four-node structure as a seed structure. Two programs for construction and cut-off are encoded rightwards on the upper ladder. At first, the reader interprets the program and the manipulator constructs an intermediate structure, which is similar to the original but without the programs. Then, a copy process follows to add the programs. Next, execution of cut-off program makes the child structure complete and separates both structures. This cut-off activates the child structure. Both develop independently thereafter. We assume that the programs are encoded on the ring as sequences of 0 and 1.

3.2 Execution of Self-replication Process

Motion of the Manipulator. Each operation is realized by an appropriate combination of the rules of graph rewriting automata.

(1) State transition operation

Execution is simple and realized only by state transition. The manipulator node M changes its state to the state corresponding to the state of Reset or Inc, and it propagates to nodes MH and MT. Then the target node updates its state.

(2) Move operation

Execution of this operation requires rearrangement of links around the manipulator. Move Right operation is realized as follows (Fig. 5). Here, three intermediate states (m_r, mt_r and mh_r) are used to represent different states of

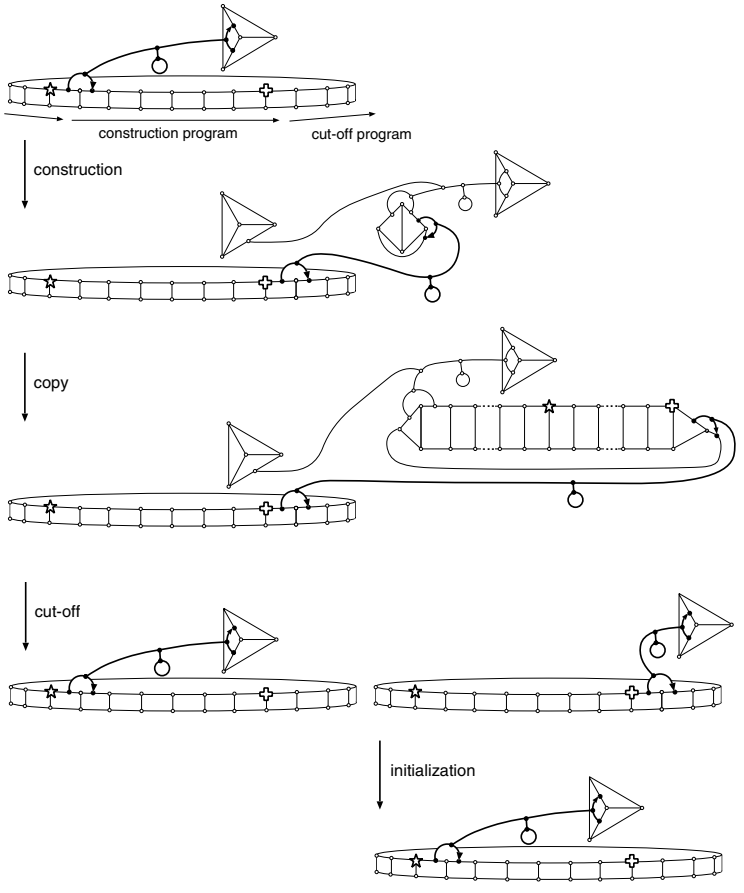


Fig. 4. Outline of self-replication process. Details of construction step are in Fig.7.

execution. When the manipulator receives Right operation from the controller, it changes the state of node M from m to m_r. It propagates to MT and MH, and changes their states into mt_r and mh_r, respectively. Then, swap rule is applied between two nodes with states m_r and mh_r to form the required structure. Finally, additional state transitions set appropriate states to the involved nodes, and completes the execution. (Note that only connection relations among nodes are in concern and their actual positions in the figures are chosen for understand-

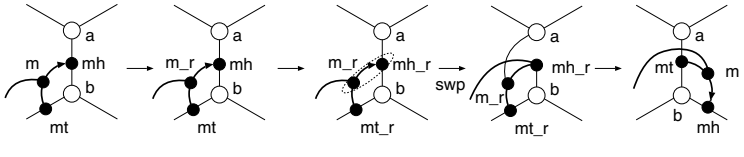


Fig. 5. Execution of Right move operation

ability.) We do not give necessary rules of graph rewriting automata explicitly, but describing them is straightforward.

Structure change for move Left operation is shown in Fig. 6. The manipulator is indicated as an arrow to simplify the figure. Internal states are not shown. Dotted lines indicate nodes to which graph rewriting rules are to be applied.

(3) Operation of executing graph rewriting rules

Four graph rewriting operations for the target node can be executed by appropriate state change and motion of the manipulator as in the move operations above. Figure 6 illustrates the steps of structure changes for each operation.

Motion of the Reader. The reader reads coded programs when requested, and transmits the codes to the controller. Appropriate treatment of the codes, including decoding, is left to the controller. Required rewriting of structures is simple and realized by simultaneous execution of com rule. (It is the same as that of the move Left operation above.)

Execution of Construction Program. Execution of the construction program generates the intermediate structure on which the programs are to be copied. Figure 7 shows the steps of construction and the corresponding sequence of operations. The reader and the programs of the parent are not shown. From the initial structure (upper left), the final structure (lower right) is constructed. This corresponds to the construction step in Fig. 4. In the resulting graph structure, the four-node structure on the left is the seed structure from the parent, and the right one is for the child. The reader is located at the Plus node after this construction.

In the figure, for simplicity, Left and Right move operations are denoted as L and R, respectively. The manipulator is designated as an arrow. Leftmost structures (A and B) on the second and the third rows are substructures of the last ones on the first and the second rows, respectively.

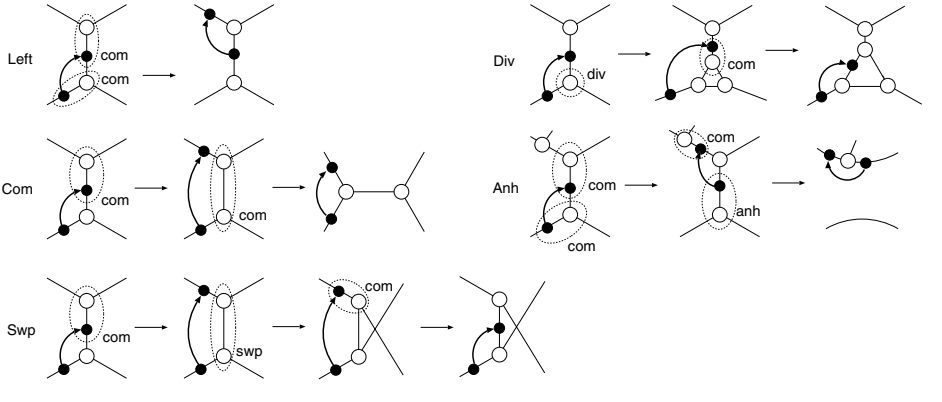


Fig. 6. Execution of four graph rewriting operations

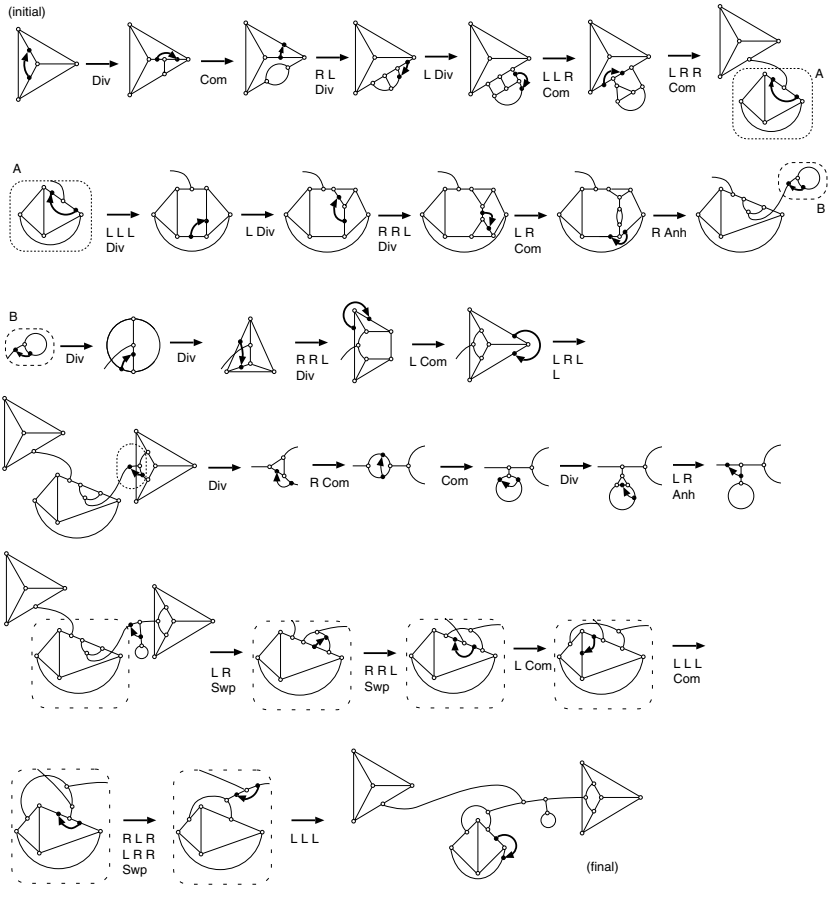


Fig. 7. Execution steps of the construction program and corresponding operations

Copy of the Program. The copy process repeats i) read the information on the ring, ii) divide the target node of the manipulator, iii) set the information on the target node, iv) move forward the manipulator to the left, until it finishes reading all the program. Figure 8 shows structure change to extend the ladder structure. We assume that appropriate set of rules is embedded. In this case, the program codes are copied without being interpreted.

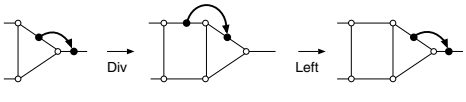


Fig. 8. Structure rewriting for copy process

Execution of Cut-off Program. The cut-off program separates the parent and the child structures, and restores the position of the manipulator to the initial position in the parent. The operation sequence of the program is as follows:

Com R Anh R L Anh R L R L Anh.

The Com operation above completes the ring structure of the child, and the last Anh actually separates two structures (see Fig. 4).

Execution of Controller. There are three kinds of modes in the self-replicating process: move, exec, and copy. In exec mode, the operations are decoded and executed. In move and copy modes, the operations are not decoded. The codes are sent to the manipulator in the copy mode. Mode transition is carried out by reading the special codes in the program: Star and Plus in Fig.4. The codes of Star/Plus makes the transition of modes respectively as: copy → copy/exec, exec → exec/copy, and move → exec/—. The controller switches these modes, and controls the execution of the reader and manipulator appropriately.

Control Flow. Program execution starts in the exec mode at the position next to the Star node in Fig. 4. The construction program constructs the intermediate structure according to the steps above. This moves the reader to the position of Plus node on the program. Reading the Plus changes the mode into copy. Then a set of copy rules are activated and the whole program information on the ring is copied, i.e., until it reads the Plus again. Reading the Plus in copy mode changes the mode into exec again. Then, the cut-off program is activated and the parent and child structures are separated. The parent structure is in the same state (location of the reader and the mode) as that of the initial, and can repeat this process. The child structure needs some post-processing because the position of the reader is next to the Plus node. This process is triggered by the separation of the structures. This separation activates the child structure in the move mode, and it skips the program until the Star, and then it changes the mode into exec. After this, the same process as that of the parent is executed.

3.3 Implementation

This self-replicating process was implemented and verified. Encoding of the operations and special codes are: L(0000), R(0001), Reset(0010), Inc(0011), Div(0100), Com(0101), Anh(0110), Swp(0111), Star(10), Plus(11). The whole program including the state setting is as follows:

Div Com R L Div L Div L L R Com L R R Com L L L Div L Div R R L Div L R Com R Anh Div Div R R L Div L Com L L R L Inc R L R Div R Com Com Div L R Anh Inc Inc Inc L R Swp R R L Swp L Com L L L Com Inc R L R L R R Swp L Inc Inc L L Plus Com R Anh R L Anh R L R L Anh Star.

The length of its codes is 380. State arrangement in the initial state of the child structure is as follows: R: 2, C1: 3, RH, MH: 1, RT, M, MT, C0: 0. Annihilation

of a link makes the nodes with state 2 and 3 adjacent. Other states are used for intermediate states during the execution. Total number of the states and the rules are 115 and 1099, respectively. Snapshots of the process are shown in Fig. 9; Figure 9(a) shows the child structure after the construction step. Each node is drawn as a triangle. Figure 9(b) is the structure several steps before the separation of the child structure.

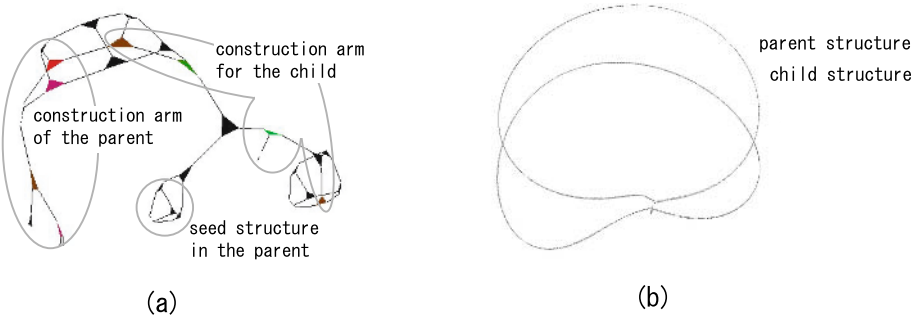


Fig. 9. Simulation result

4 Self-description for Execution

We have focused on construction. Now we consider rule execution. By introducing meta-nodes, the rule itself can be embedded in the structure in part. Such embedding permits the possibility of altering the rule and will enhance its adaptability. Note that we cannot decide the required size of internal structures in advance when the number of states can increase. Usual cellular automata are restricted on a lattice structure, and one meta-cell would have a fixed number of cells. Increasing the size of one meta-cell requires all of the other cells to increase the size at the same time. Thus, such embedding is very difficult in usual cellular automata. On the other hand, our framework permit fragments of graphs to have arbitrary many nodes inside.

One possible meta-node structure would be a hexagonal one with six nodes on exterior, three connect to outer nodes, and other three to inner structure with

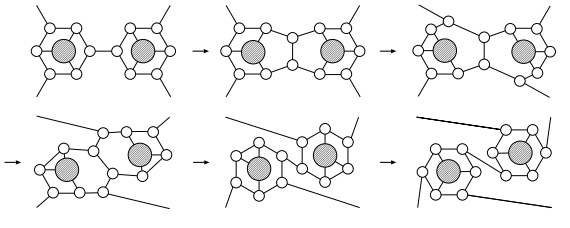


Fig. 10. Execution of commutation rule by meta-nodes

a rule set for the meta-node. The meta-node executes one graph rewriting rule by an appropriate sequence of rules on base level. For instance, commutation of meta-nodes are realized as shown in Fig. 10. In this figure, a shaded circle in each meta-node indicates the part where the program is stored. It shows only the structure change of graphs and, though indispensable, appropriate state transitions are omitted.

5 Conclusions

In this paper, we showed how self-description for construction and execution can be realized in a single framework of graph rewriting automata. In particular, we discussed self-replication in detail, and designed a sequence of operations to construct itself encoded in the structure and executed by a construction arm. This process was implemented and verified. In our formulation, operations are embedded explicitly on the graph structure, but still we rely a lot on the rules for controlling the construction arm. It is because of its simple structure. Instead, it was also possible to reduce the complication of rules by providing a graph structure in the construction arm that controls execution. Such organization is preferable for considering evolvability of the systems. It will however increase the length of construction program when we consider self-replication.

It may seem that using the construction arm dissipates in part the intricate nature of concurrency in the framework. Concurrency can be achieved by introducing multiple construction arms or naturally on the upper level as suggested in Sect. 4. Our future work includes investigation of rule rewriting strategy, and in that case it will require some external environment to be introduced. With such extensions, evolvability of systems can be envisioned. Acquisition of self-description is also an important problem in future.

References

1. Bedian, V.: Self-Description and the Origin of the Genetic Code. Biosystems **60** (2001) 39–47
2. Benko, G., Flamm, C. and Stadler, P. F.: Genetic Properties of Chemical Networks: Artificial Chemistry Based on Graph Rewriting. ECAL 2003, Lecture Notes in Artificial Intelligence **2801** (2003) 10–19
3. Rozenberg, R. (ed.): Handbook of Graph Grammars and Computing by Graph Transformation, Vol. 1: Foundations. World Scientific (1997)
4. Klavins, E.: Universal Self-Replication Using Graph Grammars. Proc. 2004 Intl Conf. on MEMS, NANO and Smart Systems (ICMENS2004) (2004) 198–204
5. Klavins, E., Ghrist, R. and Lipsky, D.: Graph Grammars for Self Assembling Robotic Systems. Proc. 2004 IEEE Intl Conf. on Robotics and Automation (ICRA2004) (2004) 5293–5300
6. McMullin, B.: John von Neumann and the Evolutionary Growth of Complexity: Looking Backward, Looking Forward.... Artificial Life **6**(4) (2000) 347–361
7. Saidani, S.: Self-Reconfigurable Robots Topodynamic. Proc. 2004 IEEE Intl Conf. on Robotics and Automation (ICRA2004) (2004) 2883–2887

8. Salzberg, C., Sayama, H. and Ikegami, T.: A Tangled Hierarchy of Graph-Constructing Graphs. Artificial Life IX: Proc. Ninth Intl Conf. on the Simulation and Synthesis of Living Systems (2004) 495–500
9. Sipper, M.: Fifty Years of Research on Self-Replication: An Overview. Artificial Life **4**(3) (1998) 237–257
10. Sipper, M. and Reggia, J. A.: Go Forth and Replicate. Scientific American **285**(2) (2001) 26–35
11. Smith, B. C.: Reflection and Semantics in Lisp. Proc. 11th ACM SIGACT-SIGPLAN Symp. on Principles of Programming Languages (1984) 23–35
12. Tomita, K., Kurokawa, H. and Murata, S.: Graph Automata: Natural Expression of Self-Reproduction. Physica D **171**(4) (2002) 197–210
13. von Neumann, J.: The Theory of Self-Reproducing Automata. Univ. of Illinois Press (1966)
14. Watson, J. D. and Crick, F. H. C.: A Structure for Deoxyribose Nucleic Acid. Nature **171** (1953) 737–738

Artificial Metabolic System: An Evolutionary Model for Community Organization in Metabolic Networks

Naoaki Ono, Yoshi Fujiwara, and Kikuo Yuta

ATR Network Informatics Laboratories,
2-2-2 Hikaridai, "Keihanna Science City",
Kyoto 619-0288, Japan
{nono, keyquo, yfujiwar}@atr.jp

Abstract. Recent studies of complex networks offer new methods for characterizing large scale of networks and provide new insights on how such networks are developed. In particular, researchers focused on biological networks such as gene regulatory systems, protein interactions and metabolic pathways in order to understand how these elemental reactions are integrated as an organism. Although various statistical features of network structures, such as scale-free or small-world, have been studied to approach underlying principles of network organization, more detailed analysis on network properties is required to understand their functions.

The community finding algorithm proposed by Girvan and Newman provides another useful technique for investigating topological structures of large networks. Applying this method to metabolic networks, we found that behavior like that of Zipf's law of the distribution of community size is shared very generally among a wide range of organisms. With the aim of realizing how this property is achieved, we present a new evolutionary model of metabolic reactions based on artificial chemistry.

1 Introduction

Recent studies of biological systems have focused on large-scale complex networks of biological interactions and found some interesting general properties such as scale-free structure and small-world [15,20] properties including gene expression [1,6], protein interaction [21,18], and metabolic pathways [19,11]. To understand the underlying principles of the emergence of those properties, various theoretical models of network evolution have been proposed [4,17,13].

On the other hand, from a biological viewpoint, a number of theories have been advanced to explain the evolution of enzyme-catalyzed metabolic networks. Horowitz proposed the retrograde model of pathway evolution [9] focusing on flows of substrates. He suggested that a new catalyst would be introduced to compensate the environmental changes and maintain metabolic flows. On the other hand, Jensen suggested a patchwork model [10] that is focusing on recruitment of enzymes across pathways. They suggest that a copy of catalyst is introduced by gene duplication at first, then, it may become to interact with a new,

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 716–724, 2005.

related substrate. Teichmann and others investigated distribution of homologs over metabolic pathways and showed that domains within the same family are widely found across different pathways [16]. Those discussions suggest that both genetic and dynamic constraints should be considered in order to understand the evolution of metabolic systems.

The purpose of this study was to investigate the effect of these evolutionary constraints on the network structures through topological analysis, in order to understand how metabolic networks are organized. One problem with analysis of metabolic networks is how to translate a stoichiometric equations to a subgraph. Arita and others indicated that properties of networks may seriously depend on the data-preparation scheme [3]. It implies that indexes such as clustering coefficients reflect only dependent features of networks rather than meaningful structures. More robust measures reflecting functional features will be required. Holme and others [8] present a method for decomposing biochemical networks into subnetworks according to the global geometry of the network. They showed interesting hierarchical subnetwork structures in metabolic pathway networks. Girvan and Newman proposed a effective algorithm for finding community structures in networks [7], which was further improved by Clauset [5] and others. Along this line, we focus on subnetwork structure analysis in order to inquire characters of network topology. We applied the Girvan and Newman's (GN) method to networks of the small molecule metabolic pathways and found that a common behavior, that is, power-law distribution of community size, is shared very widely. Next, to approach the origin of this characteristic structure, we propose an abstract model based on artificial chemistry. We introduced in this model the evolution of metabolic networks driven by gene duplication, and also considered the simple dynamics of chemical reactions to evaluate the effects of substrate-driven evolution. We show that the power-law distribution of community size is also observed in the networks evolved using this model.

2 Finding Community Structures in Metabolic Networks

The algorithm for finding subnetwork structures in networks [7] is based on the concept of "modularity". Consider a division of vertexes into some subgroups – termed "communities" – and let e_{ij} be the fraction of edges from vertexes in community i to those in community j. The modularity Q of the division which is defined as

$$Q = \sum_i (e_{ii} - \sum_j e_{ij})$$

gives a measure of whether the division is meaningful. Although true optimization of Q for a large network requires much computation, various approximate ("greedy") optimization methods are available and seem to work well. The method of optimization presented here is one of the fastest, and is described by Clauset [5]. First, let each node represent a community of size 1. ΔQ whose element ΔQ_{ij} gives the difference of Q when community i and j are combined

into one community i'. Then find maximum value of ΔQ_{ij} and combine community i and j. Repeat it until there is no pair that increases Q.

According to this method, we examined the metabolic networks based on the Kyoto Encyclopedia of Genes and Genomes (KEGG) [1]. Here are deposited metabolic pathways and an annotated genome database for hundreds of model organisms. We selected 100 of the largest organisms in order of the number of their reactions and prepared a network as follows: let p and q be the number of different compounds on each side of the equation, respectively (stoichiometric coefficients are ignored), then the equation is embedded as a complete biograph $K_{p,q}$. The average number of nodes and edges of the metabolic networks is $\bar{N}_v = 1025.0$ and $\bar{N}_e = 5368.84$. It is worth noting that the behavior of some indexes, such as clustering coefficients, seriously depends on the preparation scheme; however, we found that the structures of communities does not change qualitatively (a detailed analysis of the community structures will be reported elsewhere).

2.1 Community Size Distribution

The first quantity to be considered is the distribution of community size (i.e., the number of nodes in a community). It is reported that in some social networks the distribution appears to have a power-law form over some significant range [2]. We applied this analysis to the metabolic networks.

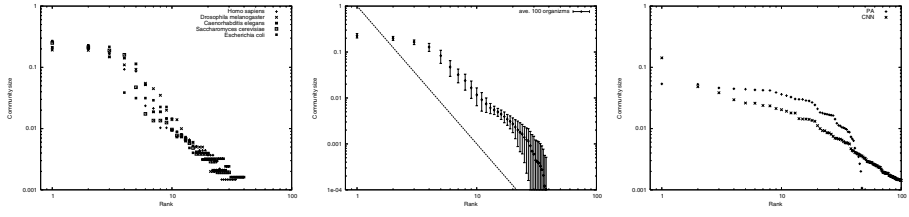


Fig. 1. Properties of metabolic networks and model networks. (a) Community size distribution of 5 popular organisms. (b) Average of community size distribution over 100 organisms. The solid line indicates a slope with exponent −3.0. (c) Community size distribution of model networks.

Figure 1(a) shows the rank-size plot of communities found in the metabolic networks of five major organisms. It is interesting that a common trend is shared by all organisms, despite the variation of their domains. Figure 1(b) shows the average of 100 organisms. There is a slow slope of large communities up to rank $r \sim 10$, followed by power-law-like decay with an exponent $2 \sim 3$. Although there seems to be a tail at the right end, i.e., some smallest communities seem to

[1] http://www.genome.jp/kegg

be larger than expected from power-law, it might be because of noise or errors; for example, since such minor compounds might be involved in some unknown or undetectable reactions, we will not discuss this aspect further at present.

To confirm that this Zipf's-law–like behavior is due not to the algorithm of community finding, but to structures of metabolic networks, we examined some models of scale-free networks in the same manner. Figure 1(c) shows the community size distribution of networks generated by methods known as preferential attachment (PA) [4] and connecting nearest neighbors (CNN) [17]. Their behavior is clearly different from that of metabolic networks. While the network of the PA model does not show power-law behavior, that of the CNN model does, although the exponent is different from metabolic networks.

3 Model: String Metabolic Network

In order to understand how this characteristic distribution of community size emerges, we here propose a model of the evolution of metabolic networks based on a framework of artificial chemistry, named String Metabolic Network (SMN). In this model, chemical compounds are represented by abstract strings and chemical reactions are recombination of the strings. A compound is an arbitrary length of string composed of eight letters, {a, b, c, . . . , h }. For simplicity, we consider only two types of fundamental reaction, as follows:

(type 1)

$$A_1A_2 + A_3A_4 \longrightarrow A_1A_4 + A_3A_2 \quad (1)$$

$$A_1A_4 + A_3A_2 \longrightarrow A_1A_2 + A_3A_4 \quad (2)$$

(type 2)

$$A_5 + A_6 \longrightarrow A_5A_6 \quad (3)$$

$$A_5A_6 \longrightarrow A_5 + A_6 \quad (4)$$

where A_i are arbitrary compound substrings. The forward and backward reactions are always coupled. A metabolic system of an "organism" is represented by two lists, a compound list containing all the strings of the chemicals appears in the cell and a reaction list corresponding to the set of enzymes encoded in its genome.

Next, we introduce evolution of the organisms. At the beginning, we start from a small set of compounds and reactions as a root ancestor. In each generation, a new reaction is appended by gene duplication and random mutation, as follows:

1. Choose a reaction from its reaction list randomly.
2. Replace a compound in the equation by a compound chosen from its compound list. If the same set of substrate already have appeared in the reaction list, back to step 1.
3. Generate a recombination rule from the new set of substrate. The reaction site, i.e., where the compound string is cut is determined randomly.

Table 1. Example of mutations

step	equation
1	ababcdcd + efefghgh ⟶ ababghgh + efefcdcd
2	ababcdcd + **ddcccbbbaa**
3	ababcdcd + **ddcccbbbaa** ⟶ ababcd**baa** + **ddcccbb**cd
1	aabccc + deefff ⟶ aabcccdeefff
2	aabccc + **ggghha**
3	aabccc + **ggghha** ⟶ aabccc**ggghha**
1	aabcccdeefff ⟶ aabccc + deefff
2	**gggbbcca**
3	**gggbbcca** ⟶ **gggbb** + **cca**

4. Append the new reaction to the reaction list.
5. If new compounds that are not in the compound list are created in step 3, append them to the list, too.

At step 4, the reverse reaction is automatically generated and appended, too. Table 1 shows examples of mutation processes.

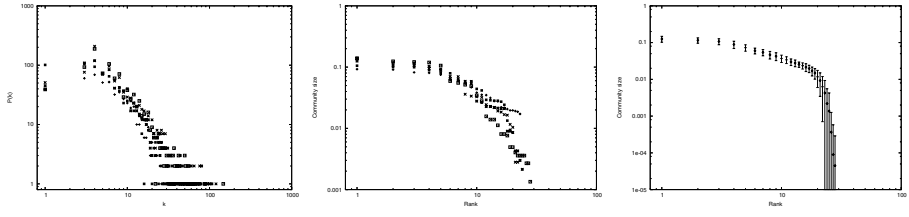
3.1 Neutral Evolution

In this paper, we introduce two different types of evolution, with or without selection pressure, according to the efficiency of the metabolic system.

First, we evolved a metabolic network without biases, i.e., we simply repeated the mutation processes 2000 times. The properties of the network generated by this evolutionary rule are shown in Fig. 2. The average is taken from 30 simulation results – 10 runs each for three different initial sets of strings and reactions (see Table 2 for more information). Because the mutation scheme of this model naturally leads to a sort of preferential attachment, the probability of a compound being involved in a new reaction is proportional to the number of the reactions in which it is already involved. Therefore, it is not surprising that

Table 2. Initial compound sets

	compound set	reaction set
set I	abcde, fghab, cdefg, habcd, efgha	abcde + efgha ⟷ abha + efgcde fghab + habcd ⟷ fghabhabcd cdefg ⟷ cde + fg
set II	ab, cde, fgha, bcdef, ghabcd	bcdef + ab ⟷ bcdb + aef fgha + cde ⟷ fghacde ghabcd ⟷ gh + abcd
set III	ba, efg, abba, addba, acbaha	acbaha + efg ⟷ acbag + efha addba + ba ⟷ addbaba abba ⟷ ab + ba



(a) (b) (c)

Fig. 2. Properties of the artificial metabolic networks. (a) Degree distribution (5 samples). (b) Community size (5 samples). (c) Community size (the average of 30 runs).

the degree distribution shows a scale-free property (Fig. 2(a)). On the other hand, although the global trends of the community size distribution seem to somehow reproduce that of living organisms, no power law behavior is observed (Fig. 2(b,c)).

3.2 Selection with Metabolic Flow

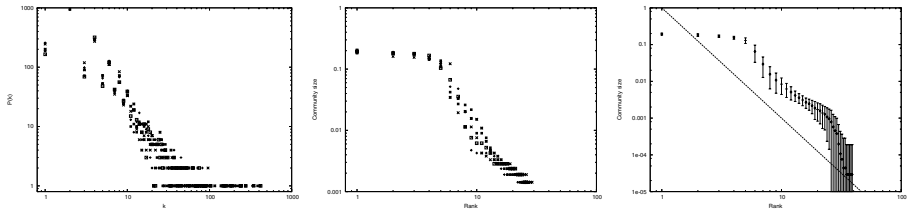
Next, we introduce flow dynamics of chemical reactions and selection for efficient metabolic network. Assume the initial set of compounds listed in Table 2 to be "resources" that are supplied from some external source. The flow rate into an organism depends on the total amount of compounds inside. Then consider that the organism synthesizes compounds from these resources through established metabolic pathways. We assume a penalty against the synthesis of larger molecules, namely, all reactions shown Equ.1–3 share a common reaction constant K, but that of Equ.4 is $0.1K$ (it corresponds to a penalty of bonding energy about $1 \sim 2kcal/mol$ at standard temperature). In general, a cell regulates the inflow/outflow of substances to keep the chemical conditions inside it stably. Although it may depend on various factors, in this model, we simply assumed that a cell maintains total amounts of substances in itself. Finally, compounds of length less than 4 escape through membranes at a constant rate K_Y. Let density of compound A at time t be $c_A(t)$ and its length be L_A and the set of resources be $\{S_R\}$, the chemical dynamics is described as follows:

$$\dot{c_A}(t) = S_A(\sum_X c_X(t)) + \sum_{X,Y} K_{XY} c_X(t) c_Y(t) - (\sum_W K_{WA} c_W(t) + \delta_A) c_A(t) \tag{5}$$

$$S_A = \begin{cases} 0 & (A \notin \{S_R\}) \\ K_S(S_0 - \sum_W c_W(t)) & (A \in \{S_R\}) \end{cases} \tag{6}$$

$$\delta_A = \begin{cases} 0 & (L_A \geq 4) \\ K_Y & (L_A < 4) \end{cases} \tag{7}$$

where K_{XY}, K_{WA} are corresponding reaction coefficients, respectively, K_S gives the resource supply rate and S_0 gives the total concentration of resources outside



(a) (b) (c)

Fig. 3. Properties of the artificial metabolic networks evolved with selection on efficiency. (a) Degree distribution (5 samples). (b) Community size (5 samples). (c) Community size (the average of 30 runs).

the cell. In the initial state, concentrations of all compounds are equal to 0 except resources whose concentration is c_0.

To select metabolic networks, we focus on "mass" of an organism, i.e., the total amount of characters in the k-th organisms which is given by $C^k(t) = \sum_X L_X c_X^k(t)$. Because of Equ. 6, the total amount of compounds always tends to converge to a constant value. Therefore, the mass of an organism represents its efficiency in synthesizing larger compounds. We evolved the network as follows:

1. Create N_μ mutants from the original organism by appending a reaction according the scheme described in the previous section.
2. Start metabolic reaction from the initial state, and after a constant time step T_G, evaluate the mass of each organisms $C^k(T_G)$.
3. Choose the organism whose $C^k(T_G)$ is the largest, and let it be the seed of the next generation.

Moreover, we add a new constraint on mutation, that is, compounds whose amounts are less than a threshold C_{cut} are ignored when we choose a new compound to replace a former entry in an equation to be mutated. In the simulation presented in this section, the parameters are given as follows: resource supply rate $K_S = 0.1$; amount of resources $S_0 = 0.2$; decay rate $K_Y = 0.1$; reaction rate $K = 0.1$; cut-off threshold $C_{cut} = 10^{-20}$.

Figure 3(c) shows the community size distribution of the results with the efficiency selection. Unlike the previous result without selection, a power-law like behavior similar to real metabolic networks can be observed.

4 Discussion and Conclusions

We have demonstrated that the analysis of the community size distribution allows us to investigate detail topology of networks. It showed the structural difference of scale-free networks whose degree distribution is quite similar. Though we reproduced a similar community size distribution, it is not clear the detail

dynamics of community organization. We are now studying some mathematical models to explain the distribution.

We have presented a model based on artificial chemistry and evolved reaction networks showing topological structure. Although there remains much to be realized, including, for example, differences of reaction coefficients, increased variety of reaction types, specific catalytic activity, we presume this model provides a useful framework for approaching the evolution of metabolic reactions.

The dominant process of metabolic enzyme evolution has been discussed to understand how the metabolic pathways were organized. There are two leading aspects were based on either substrate flow or genetic homology. For example, the retrograde evolution model is based on the former and the patchwork evolution model is later. By analyzing topology of the metabolic networks, Light and Kraulis pointed out that retrograde evolution may have played some small part while patchwork evolution seems the predominant process of metabolic enzyme evolution[12]. Our results support their observations, namely, we found that the fundamental structures of network are mainly owing to the evolutionary process, that is, new enzymes are introduced by a partial mutation of existing enzymes. While selection by the efficiency of metabolic system also plays some important part of the organization, it remains a secondary effect. Detailed investigation of the evolutionary process and quantitative analysis are required to evaluate the underlying principle of the organization of metabolic systems.

The SMN model can be further extended, for example, by introducing more realistic reaction types, the effect of chemical potential, reaction coefficients depending on the chemical types. These would be promising approaches and would be useful tools for addressing other questions about the evolution of metabolic networks, such as how ubiquitous compounds (ATP and NADP, for example) emerged.

More complex models of artificial chemistry, for example, the one based on combinator [14] has been proposed. The difference depends on their phase structures and should be studied. Evolution under different selection rules should be studied in order to understand resistance to environmental change, as well as to other properties of metabolic systems.

Acknowledgments

Dr. Shimohara of ATR Labs actively encouraged this research. This work was supported by National Institute of Information and Communications Technology (NiCT), and the project “Academic Frontier, Intelligent Information Science” (AFIIS) of Doshisha University.

References

1. H. Agrawal. Extreme self-organization in networks constructed from gene expression data. *Phys. Rev. Lett.*, 89(26):268702, 2002.
2. A. Arenas, L. Danon, A. Díaz-Guilera, P. M. Gleiser, and R. Guimerà. Community analysis in social networks. *Eur. Phys. J. B*, 38:373–380, 2004.
3. M. Arita. The metabolic world of Escherichia Coli is not small. *Proc. Natl. Acad. Sci. USA*, 101(6):1543–7, 2004.
4. A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
5. A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111, 2004.
6. D. E. Featherstone and K. Broadie. Wrestling with pleiotropy: genomic and topological analysis of the yeast gene expression network. *BioEssays*, 24(3):267–74, 2002.
7. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 99(12):7821–6, 2002.
8. P. Holme, M. Huss, and H. Jeong. Subnetwork hierarchies of biochemical pathways. *Bioinformatics*, 19(4):532–8, 2003.
9. N. H. Horowitz. On the evolution of biochemical syntheses. *Proc. Natl. Acad. Sci. USA*, 31:153–157, 1945.
10. R. A. Jensen. Enzyme recruitment in evolution of new function. *Annu. Rev. Microbiol.*, 30:409–425, 1976.
11. H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–4, 2000.
12. S. Light and P. Kraulis. Network analysis of metabolic enzyme evolution in Escherichia Coli. *BMC Bioinformatics*, 5(1):15, 2004.
13. R. Pastor-Satorras, E. Smith, and R. Sole. Evolving protein interaction networks through gene duplication. *J. Theor. Biol.*, 222:199–210, 2003.
14. P. S. di Fenizio, P. Dittrich, and W. Banzhaf. Spontaneous formation of proto-cells in an universal artificial chemistry of a planer graph. In Jozef Kelemen and Setr Sosik, editors, *Proc. the 6th European Conference on Artificial Life (ECAL'01)*, pages 206–215, Berlin, 2001. Springer.
15. S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–76, 2001.
16. S. A. Teichmann, S. C. Rison, J. M. Thornton, M. Riley, J. Gough, and C. Chothia. The evolution and structural anatomy of the small molecule metabolic pathways in Escherichia Coli. *J. Mol. Biol.*, 311(4):693–708, 2001.
17. A. Vázquez. Growing network with local rules: preferential attachment, clustering hierarchy, and degree correlations. *Phys. Rev. E*, 67:056104, 2003.
18. A. Wagner. The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes. *Mol. Biol. Evol.*, 18(7):1283–92, 2001.
19. A. Wagner and D. A. Fell. The small world inside large metabolic networks. *Proc. Roy. Soc. Lond Ser. B*, 268(1478):1803–10, 2001.
20. D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–2, 1998.
21. S. Wuchty. Scale-free behavior in protein domain networks. *Mol. Biol. Evol.*, 18(9):1694–702, 2001.

Explicit Collision Simulation of Chemical Reactions in a Graph Based Artificial Chemistry

Gil Benkö[1,2], Christoph Flamm[3], and Peter F. Stadler[2,3]

[1] Graduiertenkolleg Wissensrepräsentation
[2] Lehrstuhl für Bioinformatik, Institut für Informatik, Universität Leipzig, Haertelstraße 16-18, D-04107 Leipzig, Germany
{gil, studla}@bioinf.uni-leipzig.de
[3] Institut für Theoretische Chemie, Universität Wien, Währingerstraße 17, A-1090 Wien, Austria
{studla, xtof}@tbi.univie.ac.at

Abstract. A Toy Model of an artificial chemistry that treats molecules as graphs was implemented based on a simple Extended Hückel Theory method. Here we describe an extension of the model that models chemical reactions as the result of "collisions". In order to avoid a possible bias arising from prescribed generic reaction mechanisms, the reactions are simulated in a way that treats the formation and breakage of individual chemical bonds as elementary operations.

Keywords: Artificial Chemistry, Energy Conservation, Elementary Reactions.

1 Introduction

A chemical reaction might be regarded as a (quite arbitrarily defined) episode in the life of an aggregate of atoms. Viewing a chemical reaction as a clipping from the collection of atoms' walk through its energy landscape, which is eventually defined by quantum mechanics, we get the following picture: In the beginning, the atoms or molecules are localized in an energy well from which they cannot escape by vibrations triggered by the thermic energy. Adding thermic energy, by e.g heating the reaction vessel or radiation, enables the molecule to overcome the barriers surrounding the starting well and "hop" into one of the neighboring wells.

In many cases, the newly reached well is shallow and the molecule possesses sufficient thermic energy to escape again. These kind of wells are often called intermediary states. Eventually, the aggregate of atoms will fall into a well deep enough to be stabilized.

Artificial chemistries simulate molecules by means of matrices, strings, Turing machines, graphs or λ calculus [1,2,6,8,15,19], for a recent review see [5]. Interesting *algebraic* theories, in particular a theory of chemical organizations, have been developed based on such model. They all lack, however, a crucial ingredient featured prominently in our sketch of a chemical reaction above: they lack a natural energy function. In earlier work [3,4], we have implemented an

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 725–733, 2005.

artificial chemistry at an intermediate level of reality and computational complexity. The algorithms used here are derived from computational chemistry but simplified to a very high degree. This level of abstraction was chosen rather than more accurate or more general models for its balance between the computational tractability to guarantee a fast toy chemistry simulation and the fact that important properties of chemistry are still retained. By building on the simple Extended Hückel Theory (EHT) method [12], it naturally provides a wave function and energy as a state function. This is required for the "look-and-feel" of chemistry as a constructive system with combinatorial production of new molecules.

Knowledge-based models, in contrast, would be biased due to the intrinsic sampling bias of chemical databases as well as very expensive due to access fees. An explicit description, however, requires a natural definition of mass and, in particular for chemical reactions, energy conservation. This is ensured by representing molecules by graphs and defining energy as a state function. The difference in energy is then the driving force for chemical reactions. A more complete picture of chemical reactions must in addition include the calculation of activation barriers. Here we describe an extension to the Toy Model formalism, inspired by the ideas above, allowing the simulation of chemical reactions in line with the simplicity of the EHT method.

2 The Model

Basis

The Toy Model is derived from the 1-electron Schrödinger equation

$$\widehat{H}\Psi_\alpha = E_\alpha \Psi_\alpha \tag{1}$$

using a simplified extended Hückel theory method. A basis set consisting of $1s$ for H and $2sp^n$-hybridized Slater-type orbitals $\{\chi_i\}$ for other atoms is used to expand the molecular orbital (MO) in the form

$$\Psi_\alpha = \sum_i c_{\alpha,i}\chi_i \,. \tag{2}$$

An overlap matrix $S_{ij} = \int \chi_i \chi_j d\tau$ and Hamiltonian matrix $H_{ij} = \int \chi_i \widehat{H} \chi_j d\tau$ is set up for a molecule using fixed parameter values that are zero for non-bonded atoms. For bonded atoms, the values are tabulated according to the atoms involved, their hybridization, and their type of interaction. Valence-shell electron pair repulsion (VSEPR) theory [11] is used to determine the hybridization sp^n of an atom. Only the connectivity of the molecule is required as input.

The secular equation

$$\mathbf{H}\boldsymbol{c}_\alpha = E_\alpha \mathbf{S}\boldsymbol{c}_\alpha, \tag{3}$$

relates the matrices H and S to the wave function described by the vector of coefficents $\boldsymbol{c}_\alpha$. Solving eq. 3 yields the energy E_α of molecular orbital α and the

wave function. In principle, any molecular property might be computed from these values. In particular, we calculate the spectrum, charges, energy, and, in combination with a spectral embedding, the dipole and solvation energy. The Toy Model comes with parameter values for C, H, N, O, P, and S, and can be easily extended by editing the parameter file. The calculation takes into account σ bonds, π bonds, backbonding and hyperconjugation through indirect sp^n-sp^n and sp^n-p overlaps, banana overlaps in rings, and stronger backbonding in P and S. In line with [9] d orbitals are seen as polarization functions, which we "replace" in our simplified framework by stronger backbonding in P and S.

Extension to Chemical Reactions

We extended the Toy Model to chemical reactions by decomposing a reaction into small "moves": the formation, breaking, or shifting of bonds, in analogy to the elementary moves of the Dugundji-Ugi model [6,16]. We took into account both single bond formation and simultaneous formation of two bonds, as there is for example controversy about whether the Diels-Alder reaction (see below) proceeds in a concerted fashion (simultaneous bond formation) or via a diradical (sequential bond formation). The same applies for single and double breaking of bonds. Finally, also the shifting of a bond was included in our simulation. The sequence of moves is determined by a simulation based on a continuous time Monte Carlo method proposed in [10]:

```
1: if reaction is monomolecular (one reacting molecule) then
2:    find neighbors produced by single/double formation, single/double breaking, shifting
3: end if
4: if reaction is bimolecular (two reacting molecules) then
5:    find neighbors produced by single/doubleform, shifting
6: end if
7: loop
8:    if all neighbors are of higher energy then
9:       stop
10:   end if
11:   move to neighbor with lowest energy
12:   if this neighbor is a split molecule then
13:      stop
14:   end if
15:   find neighbors produced by single/double formation, single/double breaking, shifting
16: end loop
```

Interaction of Orbitals

We now describe how the selection of bonds that may form during a move was restricted. Atomic orbitals were divided into four types in view of possible interactions and thus bond formations: the sp^3 orbital as it features a particularily

Table 1. Table of allowed interactions

Atom hybridization and type of orbital 2		Atom hybridization and type of orbital 1						
		sp^3		sp^2		sp		s
		b	l	p	l	p	l	
sp^3	b		•		•		•	
	l	•		•		•		•
sp^2	p		•	•		•		•
	l	•						•
sp	p		•	•		•		•
	l	•						•
s			•	•	•	•	•	

accessible back lobe (b), the s orbital, the p orbital, and the orbital occupied by a lone electron pair or lone electron (l). The b type is important for its role in the S_N2 reaction (nucleophilic substitution): there a l type orbital interacts with the back lobe of a sp^3 orbital and eventually creates a new bond.

We further define that only the interactions $p-l$, $l-s$, $p-p$, $p-s$ and $b-l$, shown in Fig. 1, are possible between the orbital types. This "sophisticated guess" is based on the importance of those interactions in common organic reactions, as shown in any standard organic chemistry textbook [20] and in Orbital Interaction Theory [7,14,17]. In Fig. 1(right column), we show for example the interactions in the hydroboration, the $E1$ elimination, the Diels-Alder, the hydrogen shift, and the aforementioned S_N2 substitution reaction.

The possible interactions according to the hybridization of the two interacting atoms are summarized in Tab. 1. Fig. 2 shows some allowed interactions for an example pair of molecules.

The bonds that may break are not restricted to special types. However, we take into account that bond fission can be homolytic or heterolytic. Thus there are three cases: (1) the two fragments each inherit one electron of the bond (radical formation); (2) the first fragment receives both electrons; (3) the second fragment received both electrons (ion formations). The shifting of a bond is simulated in our model by simultaneous fission (unrestricted) and formation (restricted by the allowed interactions above) of a bond.

3 Examples

Chemical reactions were simulated for two sets of reacting molecules: first, ethene and butadiene, and second, the enolate of ethanal (acetaldehyde). We generated all molecular configurations (products) accessible from the reactant(s) within our framework, and whose energies were lower than or within 3 kcal/mol of the energy of the reactant(s).

For ethene and butadiene, both molecules can react by themselves (mono molecular reactions), with another specimen of the same type, or with each other (bimolecular reactions) (Tab. 2).

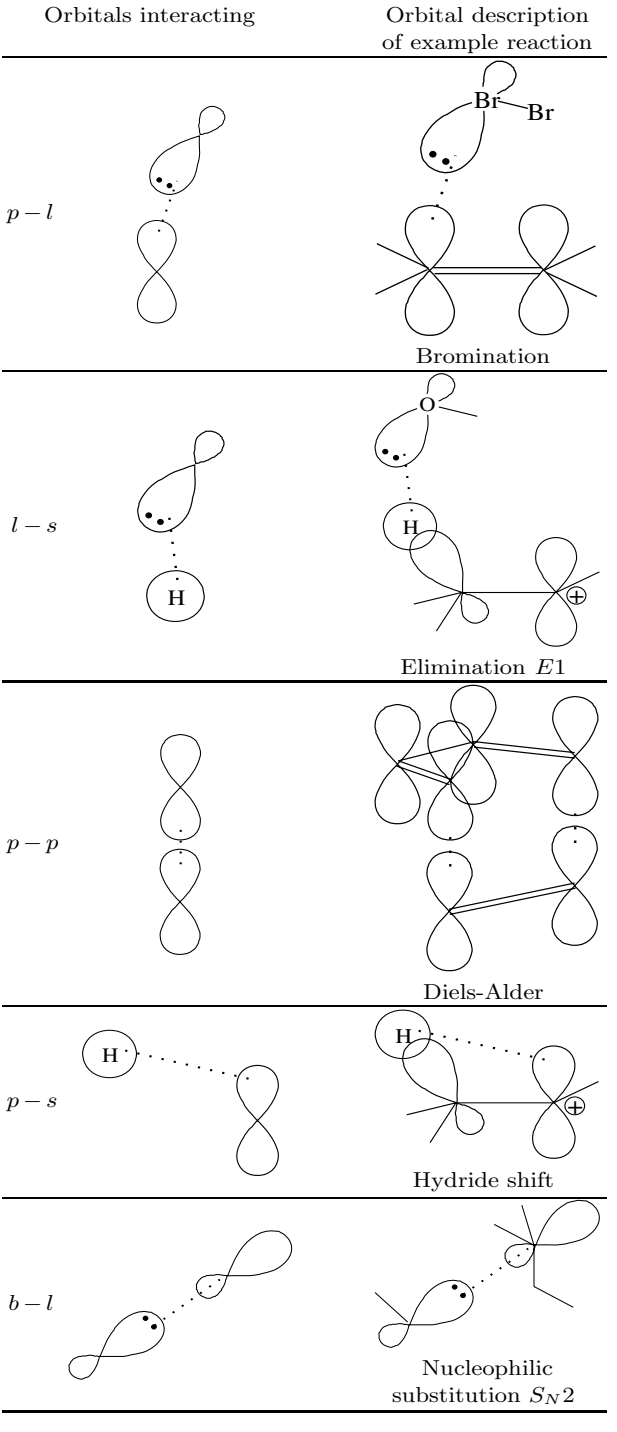


Fig. 1. Allowed interactions. Sketches follow [20].

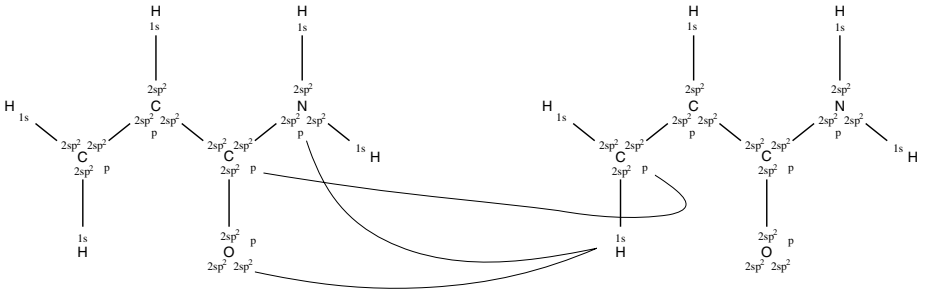


Fig. 2. Selected allowed interactions between two propenamide molecules. The molecules are shown with their atoms surrounded by their valence orbitals. Curved lines indicate some of the allowed interactions: *p-p*, *p-s*, and *p-l* (the sp^2 orbital on O is occupy lone pair and thus a *l*-type orbital).

Table 2. Reactions of ethene and butadiene. Molecules are written in the SMILES notation [21], which is very similar to a structural formula. Diels-Alder educts are indicated by (DA). Energies are total atomization energies.

	Structural formula	Energy (kcal/mol)
Reactant	C(H)(H)=C(H)(H)	-415.95
Products	none (no reaction)	
Reactant	C(H)(H)=C(H)C(H)=C(H)(H)	-789.62
Products	C(H)(H)(CC(H)=C(H)H)H	-787.86
(of higher	C(H)(H)(C(H)[C-]=C(H)H)[H+]	-787.88
energy)	C(H)(H)([C-]C(H)=C(H)H)[H+]	-787.86
Reactants	C(H)(H)=C(H)(H) + C(H)(H)=C(H)(H)	-831.90
Products	none (no reaction)	
Reactants	C(H)(H)=C(H)C(H)=C(H)(H) + C(H)(H)=C(H)C(H)=C(H)(H)	-1579.24
Products	C1(H)(H)C(H)C(H)C(H)(H)C(C(C(C1(H)H)H)H)(H)H (DA)	-1588.74
	C(H)(H)=C(H)C1(H)C(H)(H)C(C(C(C1(H)H)H)H)(H)H (DA)	-1587.39
Products	C(H)(H)(C(H)C(H)=C(H)H)[C+](C(H)H)C(H)=C(H)H + [H-]	-1577.75
(of higher	C(H)(H)[C+](C(H)=C(H)H)C(H)(H)C(H)C(H)=C(H)H + [H-]	-1577.83
energy)	C(H)(H)[C+](C(H)=C(H)H)C(C(C(=C(H)H)H)H)(H)H + [H-]	-1577.42
	C(H)(H)=[C-]C(H)=C(H)H + [H+]C(H)(H)C(H)C(H)=C(H)H	-1577.49
Reactants	C(H)(H)=C(H)(H) + C(H)(H)=C(H)C(H)=C(H)(H)	-1205.57
Product	C1(H)(H)C(H)(H)C(C(C(C1(H)H)H)H)(H)H (DA)	-1222.33

As expected, one of the reactions occuring is the formation of a ring. This pericyclic [4+2]-cycloaddition reaction between a diene and an alkene is called the Diels-Alder (DA) reaction. The DA reaction is thermally allowed since the HOMO of the butadien is of the same symmetry as the LUMO of the alkene. The model reproduces the correct behavior between butadien and ethene. In addition, the model correctly predicts that the thermal [2+2]-cycloaddition reaction between two alkenes (in our case ethenes) to a cyclobuten derivate does

Table 3. Reactions of the enolate of ethanal. Aldol condensation products are indicated by (ac).

	Structural formula	Energy (kcal/mol)
Reactant	[C-](H)(H)C(H)=O	-667.48
Product	[C-](H)(H)([C-]=O)[H+]	-676.65
Reactants	[C-](H)(H)C(H)=O + [C-](H)(H)C(H)=O	-1334.95
Products	[C-](H)(H)[C-]=O + [H+][C-](H)(H)C(H)=O	-1344.14
	[C-](H)(H)(C(H)=O)[C+]([C-](H)H)O + [H-]	-1343.72
	[C-](H)(H)[C+](O)[C-](H)(H)C(H)=O + [H-]	-1343.88
	[C-](H)(H)C(H)(O)[C-](H)(H)C(H)=O (ac)	-1336.94
	[C-2](H)C(H)=O + [H+][C-](H)(H)C(H)=O	-1335.50
Products (of higher energy)	[C-](H)(H)(C(H)=O)C(H)C(H)=O + [H-]	-1332.64

not occur, since this reaction is symmetrically forbidden. There are, however, several examples of [2+2]-cycloadditons in the literature which are believed to occur through a radical or dipolar intermediate.

The reactants alone, as well as a collision of two ethene molecules, do not form products of lower energy, i.e., they do not react without energy input from outside (first three parts of Tab. 2). The kinetic energy or the energy stored in molecular rotation or vibration may however suffice depending on the temperature to attain products of higher energy. The isomers of butadiene within 3 kcal/mol of the ground state are hydrogen shifts and are artifacts of the model and its parameters which favor sp^3 over sp^2 atoms.

On the other hand, butadiene can react with either ethene or another butadiene (last two parts of Tab. 2). It forms rings by two simultaneous bond formations bridging twice the gap between two molecules. This Diels-Alder reaction has two possible outcomes for the reaction of butadiene with itself (regioisomers). Again, products of higher energy include artifacts like hydrogen shifts or formation of hydride.

The condensation reaction of enolate anions and carbonyl derivates, one of the most useful reactions in organic chemistry, is as well correctly rendered by our model. We simulated the case for the enolate of ethanal (Tab. 3). This reaction is termed Aldol Condensation (ac) and is an acyl addition reaction of the nucleophilic enolate to the electrophilic carbonyl carbon. However, here also the other products of lower energy are artifacts (hydrogen shift or hydride formation).

The artifacts in the two examples above are results of the particular parameter set, which is taken from the literature. For example, the values S_{ij} and H_{ij} for sp^2 hybridized carbons are based on experimental bond lengths. Adjusting them in order to improve the results only produced more artifacts. The artifacts might also result from underestimating the electronic repulsion between charges in the molecule, or neglecting the energy involved in the separation of charges. Future implementation will take these effects into account.

4 Concluding Remarks

The model presented here tries to build a formalism of chemistry with the least possible bias. To this end, we choose a simple molecule representation using graphs which still allows an energy calculation. On the other hand, we use a rather drastic abstraction from reality. While the approach is still based on quantum chemistry, it avoids e.g. the complications of spatial embedding in order to keep the resource requirements tractable even for large scale simulations. Chemical reactions are implemented here not in the forms of prescribed rewrite rules as in [3,4]. Instead, the model relies on a decomposition into bond formations, fissions, and shifts. The result is an artificial chemical reaction which reproduces some of the experimental results and features energy dependence, reaction specificity, and multiple possible reaction outcomes.

In the present implementation, we only looked at the thermodynamic description of a chemical reaction. In future work, we intend to include the kinetic constraints, in particular by incorporating the reaction barriers calculated using the Klopman-Salem equation [13,18]. Since our basic model included also the calculation of solvation energies, we will also take into account solvation effects on chemical reactions.

The simulation results show that our model is indeed capable of producing the chemically expected reaction pathways despite drastic simplifications of this artificial chemistry relative to "real" quantum chemistry. Due to these simplifications, artifacts (relative to reality) are to be expected, and indeed do occur. However, these artifacts still conform to the 'look-and-feel' of chemistry, and could be avoided by choosing different parameters — probably at the expense of opening up other reaction pathways that do not occur in reality. We emphasize, that our model is not intended as a replacement of quantum-chemical computations for the predictions of properties and reactions of real molecules. Rather, our goal is a computationally tractable *artificial chemistry* model, in which not all combinatorially possible reactions occur indiscriminately, but depend on an energy state function. In this setting multiple reaction outcomes are possible so that different reaction channels in general have different rates that self-consistently determined from within the model.

Acknowledgments

This work was supported by the Graduiertenkolleg Wissensrepräsentation, Universität Leipzig, COST Action D27, and the Bioinformatics Initiative of the DFG.

References

1. R. J. Bagley and J. D. Farmer. Spontaneous emergence of a metabolism. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, Santa Fe Institute Studies in the Sciences of Complexity, pages 93–141, Redwood City, CA, 1992. Addison-Wesley.

2. W. Banzhaf, P. Dittrich, and B. Eller. Self-organization in a system of binary strings with spatial interactions. *Physica D*, 125:85–104, 1999.
3. Gil Benkö, Christoph Flamm, and Peter F. Stadler. Generic properties of chemical networks: Artificial chemistry based on graph rewriting. In W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, editors, *Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life (ECAL)*, pages 10–20. Springer, 2003.
4. Gil Benkö, Christoph Flamm, and Peter F. Stadler. A graph-based toy model of chemistry. *J. Chem. Inf. Comput. Sci.*, 43:1085–1093, 2003.
5. Peter Dittrich, Jens Ziegler, and Wolfgang Banzhaf. Artificial chemistries — a review. *Artificial Life*, 7:225–275, 2001.
6. J. Dugundji and Ivar Ugi. Theory of the *be*- and *r*-matrices. *Top. Curr. Chem.*, 39:19–29, 1973.
7. Ian Fleming. *Frontier Orbitals and Organic Chemical Reactions*. John Wiley, 1976.
8. Walter Fontana. Algorithmic chemistry. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 159–210, Redwood City, CA, 1992. Addison-Wesley.
9. D.J̃. Gilheany. No d orbitals but walsh diagrams and maybe banana bonds: Chemical bonding in phosphines, phosphine oxides and phosphonium ylides. *Chem. Rev.*, 94:1339–1374, 1994.
10. Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81:2340–2361, 1977.
11. R.J̃. Gillespie and R.S̃. Nyholm. Inorganic Stereochemistry. *Quart. Rev. Chem. Soc.*, 11:339–380, 1957.
12. Roald Hoffmann. An Extended Hückel Theory. I. Hydrocarbons. *J. Chem. Phys.*, 39(6):1397–1412, 1963.
13. G. Klopman. Chemical reactivity and the concept of charge- and frontier-controlled reactions. *J. Am. Chem. Soc.*, 90:223–243, 1968.
14. G. Klopman. *Chemical Reactivity and Reaction Paths*. Krieger, 1974.
15. John S. McCaskill and Ulrich Niemann. Graph replacement chemistry for DNA processing. In A. Condon and G. Rozenberg, editors, *DNA Computing*, volume 2054 of *Lecture Notes in Computer Science*, pages 103–116. Springer, Berlin, D, 2000.
16. G. Nowak. Common-sense reasoning cast over D-U model in simulation of chemical reactions. *Int. J. Quantum Chem.*, 84(2):282–289, 2001.
17. Arvi Rauk. *Orbital Interaction Theory of Organic Chemistry*. Wiley-Interscience, 2000.
18. L. Salem. Intermolecular orbital theory of the interaction between conjugated systems. I. General theory; II. Thermal and photochemical calculations. *J. Am. Chem. Soc.*, 90:543–552 & 553–566, 1968.
19. Marcel Thürk. *Ein Modell zur Selbstorganisation von Automatenalgorithmen zum Studium molekularer Evolution*. PhD thesis, Universität Jena, Germany, 1993. PhD Thesis.
20. K. P.C̃. Vollhardt and N. Schore. *Organic Chemistry*. W. H. Freeman, 4th edition, 2002.
21. D. Weininger. SMILES, a chemical language and information system. *J. Chem. Inf. Comput. Sci.*, 28:31–36, 1988.

Self-replication and Evolution of DNA Crystals

Rebecca Schulman and Erik Winfree

California Institute of Technology, Pasadena, CA 91125, USA
{rebecka, winfree}@caltech.edu

Abstract. Is it possible to create a simple physical system that is capable of replicating itself? Can such a system evolve interesting behaviors, thus allowing it to adapt to a wide range of environments? This paper presents a design for such a replicator constructed exclusively from synthetic DNA. The basis for the replicator is crystal growth: information is stored in the spatial arrangement of monomers and copied from layer to layer by templating. Replication is achieved by fragmentation of crystals, which produces new crystals that carry the same information. Crystal replication avoids intrinsic problems associated with template-directed mechanisms for replication of one-dimensional polymers. A key innovation of our work is that by using programmable DNA tiles as the crystal monomers, we can design crystal growth processes that apply interesting selective pressures to the evolving sequences. While evolution requires that copying occur with high accuracy, we show how to adapt error-correction techniques from algorithmic self-assembly to lower the replication error rate as much as is required.

1 Introduction

It is widely accepted that Darwinian evolution is responsible for the complexity and adaptability seen in modern biology. However, the mechanisms by which evolving organisms adapt to their environment are not well understood. An important roadblock in studying evolution is the dearth of physical systems in which evolution can be studied; a tractable synthetic system for replication and evolution would facilitate the study of how physical selection pressures lead to evolutionary adaptation. A chemical self-replicator might also be used to evolve solutions to problems in chemistry or nanotechnology. If such a system were simple enough, it could also shed light on how self-replication emerged spontaneously at the origin of life.

In 1966, Graham Cairns-Smith proposed a simple mechanism by which polytypic clay crystals could replicate information in the absence of biological enzymes [3,4]. Polytypic clay crystals are crystals where the orientations of subsequent layers can differ, and therefore a cross-section of the crystal contains an information-bearing sequence. Crystal growth extends the layers and copies the sequence of orientations, which may be considered its genotype. Occasionally, physical forces break a crystal apart. Because crystals replicate their genotype many times during growth, splitting of a crystal can yield multiple pieces, each containing at least one copy of the entire genotype. Cycles of growth and fragmentation cause each sequence to be exponentially amplified.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 734–743, 2005.

We propose a method of self-replication that works by similar growth and fragmentation of algorithmic DNA crystals. DNA crystals are composed of DNA tile monomers [8]. Different types of DNA tiles can be designed to assemble via programmable rules [18]; a typical DNA crystal is assembled from several tile types. As in Graham-Smith's conception, DNA crystals can contain a sequence that is copied during growth, in this case a linear arrangement of DNA tile types (Figure 1a). Unlike most types of clay crystal growth, DNA crystal growth is tractable in the laboratory and occurs at time scales (hours) that are suitable for experimental investigation.

It is perhaps surprising that DNA crystal replication exhibits many of the phenomena of interest for the study of Darwinian evolution. In Section 2, we describe in more detail how crystal evolution works and introduce the components of DNA crystals and a model of the growth process. The examples in Sections 3 and 4 illustrate how DNA crystals can copy arbitrary amounts of information and how in particular environments, this information affects the replication rate. In Section 5, we describe techniques for increasing the accuracy of replication.

2 Replicating Information with DNA Crystals

DNA crystals consist of DNA tile monomers [8] which can attach to other tiles in a programmable fashion: each of the four sides of the DNA tile has a short single stranded portion which can hybridize with the complementary strand of another tile (Figure 1b). DNA tiles can assemble into 2-dimensional crystals [21] and can

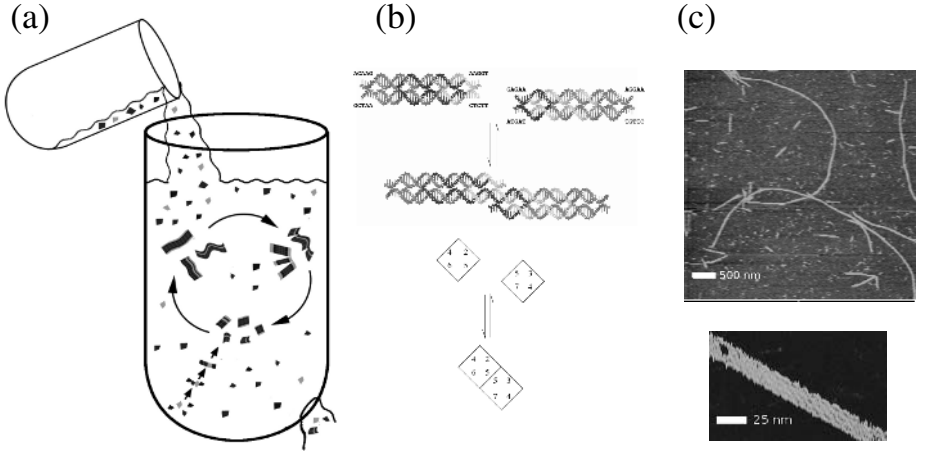


Fig. 1. DNA crystals. (a) The DNA crystal life cycle. The materials required for growth are constantly replenished. Crystals die when they are flushed out of solution in an exit stream. **(b)** Tiles with complementary single-stranded sticky ends can attach by hybridization. For convenience, DNA tiles may be represented as square tiles; tiles with the same side labels correspond to molecules with matching sticky ends. **(c)** Atomic force microscopy image of DNA crystals formed by the molecules shown in Figure 2b. At higher resolutions, individual tiles can be discriminated.

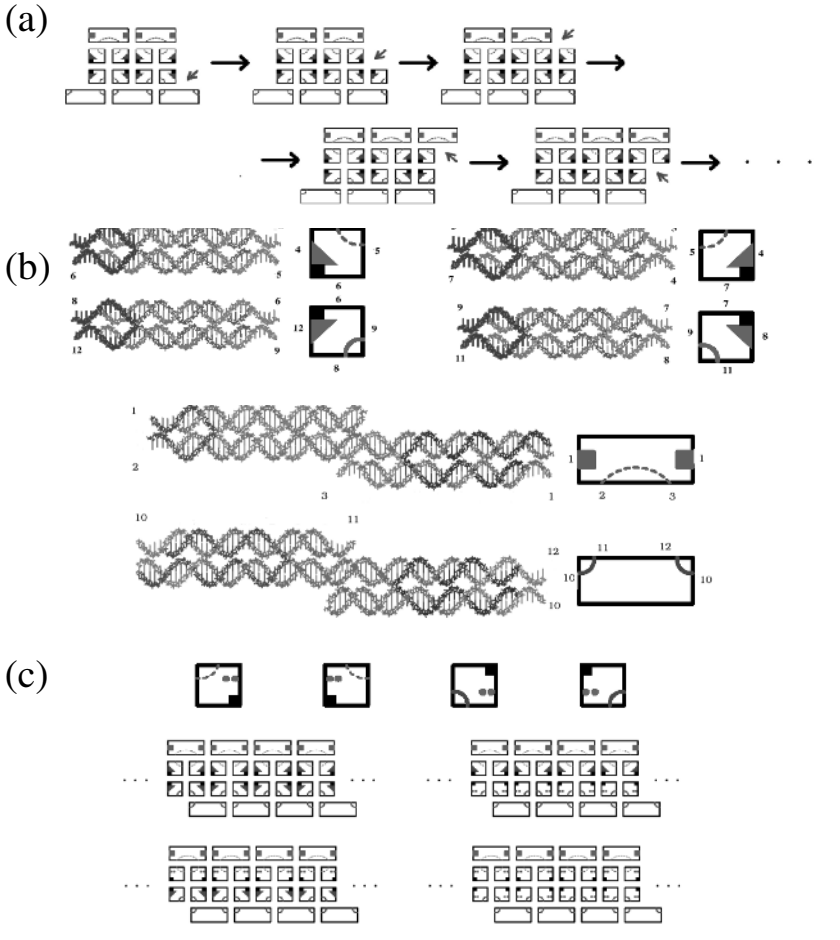


Fig. 2. The zig-zag tile set. (a) A zig-zag assembly. Two alternating tile types in each row enforce the placement of the double tiles on the top and bottom, ensuring that under algorithmic assembly conditions, growth occurs in a zig-zag pattern. Although only growth on the right end of the molecule is shown here, growth occurs simultaneously on both ends of the molecule. At each step, a new tile may be added at the location designated by the small arrow. **(b)** The basic zig-zag tile set consists of six molecules (tile types). Each square and rectangle shown is a logical representation of the molecule shown to its left. By convention, tiles cannot be rotated. The tiles shown here have unique bonds that determine where they fit in the assembly: each label has exactly one match on another tile type. While the logical representations of DNA tiles have the same connectivity as DNA tile molecules, the logical representation of a tile has a different aspect ratio and labels in different orientations than the actual molecules. **(c)** The tile set shown in Figure 2b forms only one type of assembly. A tile set consisting of the tiles in (b) and the four tiles shown here allows four types of assemblies to be formed. The vertical column of each type contains a different 2-bit binary sequence.

be programmed to form other structures, such as thin ribbons (Figure 1c). A wide variety of DNA tile crystals have been synthesized [10,15,5].

Under algorithmic assembly conditions [19], the assembly of DNA tiles into a crystal is only energetically favorable when it occurs cooperatively, i.e. by the formation of two or more sticky end bonds. The attachment of a tile to a crystal performs a step of a computation in the sense that a unique tile (among many possible in solution) may attach at a particular growth location. With an appropriate choice of tiles, DNA tile assembly can perform universal computation [18,2].

The zig-zag crystal shown in Figure 2a is formed from the tiles shown in Figure 2b. Matching rules determine which tile fits where. When a zig-zag crystal is added to a solution of free tiles under algorithmic assembly conditions, growth is constrained to occur in a zig-zag pattern by the requirement that each tile addition must form two or more sticky end bonds, as shown in Figure 2a. It is easy to confirm that under such conditions, there is always a unique tile that may be added on each end of the ribbon.

Zig-zag crystals are designed so that under algorithmic assembly conditions, growth produces one new row at a time, and continued growth repeatedly copies a sequence. The requirement that a tile must attach by two bonds means that it must match both its vertical neighbor (another tile that is part of the new column being assembled), and its horizontal neighbor (in a previously assembled row). Several tiles might match the label on the vertical neighbor, but because tiles must make two correct bonds in order to join the assembly, only a tile that also matches the label on the horizontal neighbor can be added. Therefore, the tile being added in the new column must correspond to the one in previous column. As a result, information is inherited through templated growth. The set of tiles formed by adding the tiles in Figure 2c to those shown in Figure 2b can propagate one of four strings. Additional tiles may be added to the set of tiles in Figures 2b and 2c to create a tile set that copies one of 2^n sequences of width n. We will later discuss tile sets in which an unbounded amount of information can be copied.

The growth of a zig-zag DNA crystal increases the number of copies of the original information present in the ribbon, but does not change the rate at which new copies of the sequence are produced. The rate of copying can be sped up by shear forces that cause crystals to break. With each new crystal that is created by breakage, two new sites become available to copy information. Repeated applications of shear force interspersed with time to grow therefore exponentially amplify an initial piece of information. Occasionally, a tile matching only one bond rather than two will join the assembly, resulting in occasional copying errors, which are also inherited. If errors happen during copying, which they will under almost any achievable condition [19], and crystals with particular sequences grow faster than others, then evolution can occur.

3 The Royal Road

A selection experiment for DNA crystal evolution involves both an environment (available resources and laws of chemistry and physics) and DNA crystals that

grow and reproduce within that environment. Artificial evolution experiments must set up both. Here, a set of DNA tiles is used to define an environment for crystal growth. The set of DNA tiles determines the set of sequences which may be copied and the "chemistry" of the system, i.e. the rules which tiles bind to each other [1]. A particular arrangement of DNA tiles is the information that is propagated in these experiments, the genotype; it is the organism being evolved. The phenotype of a sequence is its replication rate in the given environment. In this section we describe a tile set that allows many kinds of sequences to grow; a selection pressure results from physical conditions in which tile concentrations differ for each tile type.

A DNA crystal can grow only when it comes in contact with a tile that can be added favorably to the crystal. In a well-mixed reaction vessel, the higher the concentrations of tiles of the type that may be legally added, the more quickly such contact occurs. Therefore, a simple selection pressure results from a difference in concentration between tile types used to copy the sequence information: assemblies with sequences containing tiles present at high concentrations will grow and reproduce faster than assemblies with sequences containing tiles present at very low concentrations.

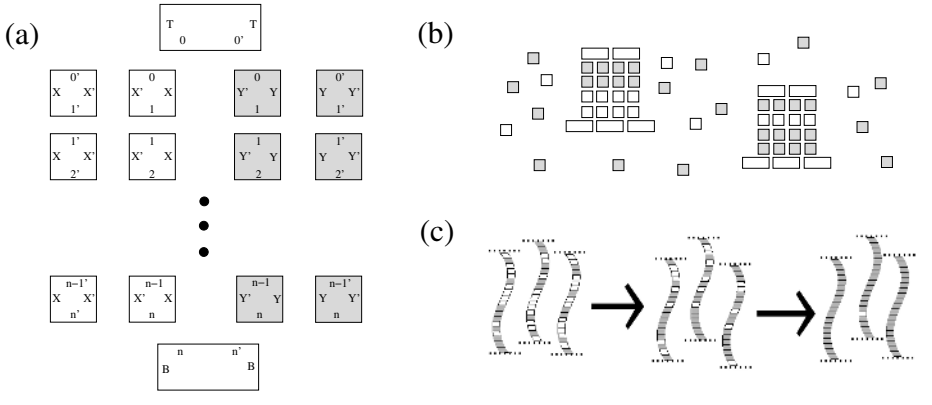


Fig. 3. The royal road tile set. (a) The royal road tile set consists of four tiles for each of n sequence positions, two for propagating an X bit and two for propagating a Y bit. Two boundary tiles are also used. 2^n different sequences can be copied with this tile set. **(b)** When more Y than X tiles are present, sequences containing more Y tiles tend to grow faster. **(c)** As growth progresses, sequences containing mostly Y tiles become more and more common. Each sequence shown represents an assembly consisting of many copies of the illustrated sequence.

[1] Our choice of terminology reflects the observation that whether a self-replicator is made from clay, biological polymer or other material, the chemistry of the specific elements involved determines the evolutionary landscape. As an example, the chemistry of nucleic acids can make some sequences hard to copy. Certain sequences fold up or bulge [9], making copying of those sections more difficult. Here, the constraints are not on how a sequence folds, but on how its elements fit together: the tile set similarly determines the evolutionary landscape.

A tile set in which one of two bits can be propagated at each of n sequence positions is shown in Figure 3a. Let X_i and Y_i be the two tile types that can be propagated at position i. If Y_i is present in solution at a concentration higher than that of X_i, as in Figure 3b, the fitness landscape for this selection resembles the simplest case of a well-studied problem in genetic algorithms, the "royal road" [12]. Here, the growth rate increases monotonically with the number of Y_i's in the sequence s. So long as the Y_i tiles remain more common in solution, sequences containing only Y_i tiles will quickly dominate (Figure 3c).

4 Selection of Regular Languages

Section 3 illustrated how tile concentration can create a selection pressure, causing some sequences to grow faster than others. While this is a simple selection pressure to understand, the adaptation that occurs is also simple. In this section we describe how a single tile set allows for the replication of an infinite number of sequences and how sequence constraints imposed by the tile set can provide more interesting selection pressures.

In the previous example, the "chemistry" of the tile set determined the length of sequences that could be copied, and which tiles could be used in which position of the sequence. Here we consider evolution when the tile set "chemistry" allows only certain sequences to be copied, but they may be arbitrarily long. In particular, the tile set environment allows copying only of sequences that are accepted by a particular finite state machine.

A finite state machine is an abstract device that can perform a computation requiring only a fixed amount of memory. It consists of a set of states and rules describing how to transition between states as each character of input is received. Computation begins in a prescribed state. When the inputs have all been received, the current state is either in an accept state, in which case the the input is accepted, or a reject state. Figure 4a shows a simple finite state machine (along with the tiles that implement the transition steps of the machine) which detects whether the number of ones in a binary sequence input is divisible by three.

The self-assembly of DNA sequence [1] and tile [13] alphabets can generate the set of sequences accepted by a given finite state machine, also known as a regular language. Accepted sequences can be of any length. In contrast to the tile sets described in Section 3, where the top and bottom sides of a tile encode the position in the fixed-length sequence where the tile can be added, the top and bottom sides of the tiles in Figure 4a encode the state of the machine as it processes each character of the sequence being copied.

A tile set that copies only inputs accepted by a given finite state machine is constructed as follows. Each possible transition between states is encoded as a single tile (Figure 4a). The left and right sides of the tile encode the input, the top side encodes the state that machine is in before the input is received and the bottom side encodes the state that the machine transitions to after the input has been received. The top boundary tile encodes the start state and a bottom boundary tile encodes each accept state (Figure 4b). Another set of tiles copies a sequence that has been accepted by the machine. These tiles have only one

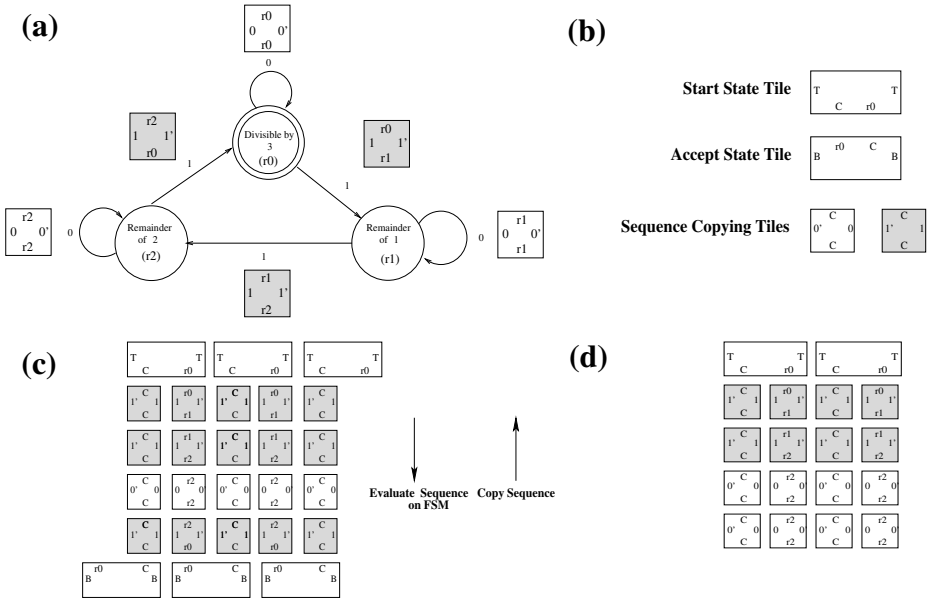


Fig. 4. Selection of sequences with particular numbers of logical 1's. (a) A diagram of a finite state machine that can determine whether a binary sequence contains a number of 1's that is divisible by 3. The double circled state is both the start and the accept state. (In general these states are not the same.) The tiles shown can be used with the tiles in (b) to follow the instructions of the machine during tile assembly. **(b)** Additional tiles needed to complete the tile set in (a). The construction shown in (a) and (b) can be generalized to any finite state machine. **(c)** An assembly encoding a sequence accepted by the machine in (a). Evaluation ends in an accept state, so a bottom tile may be added and assembly can continue. **(d)** An assembly encoding a sequence not accepted by the finite state machine in (a). Because execution of the finite state machine ended with a state other than the accept state, assembly cannot continue.

state on their bottom and top sides, and encode the same sequence bit on their left and right sides.

During growth down the crystal[2], assembly evaluates the sequence according to the finite state machine's rules. If the machine ends in an accept state, a bottom tile can bind to the site and upward growth can begin (Figure 4c). If the machine is not in an accept state, no bottom tile exists which matches the

[2] Growth on the left side of the zig-zag crystal in Figure 4c reads the sequence elements backward, and evaluates the finite state machine in reverse. While running the finite state machine shown in Figure 4a backward accepts the same set of states as running the machine forward, for other machines there may be non-determinism when the machine is run in reverse. A step may be possible that cannot lead to the start state, leaving an uncompleted assembly. Assemblies corresponding to tile sets of this type will grow mostly in the direction where the finite state machine is evaluated in the correct direction. With some additional complexity, it is also possible to replace this tile set an equivalent tile set that can grow only in the forward direction [17].

growth front, and growth stops (Figure 4d). Thus, only sequences which are accepted by the machine will continue to be replicated. These sequences will be the ones that are selected for.

More complex selection pressure results if the crystals grown in this tile set environment are moved to an environment containing tiles that accept a different language of sequences. For example, crystals grown using the tiles shown here might be moved to a mixture containing tiles that allowed only sequences with a number of ones that is divisible by 5 to grow. Only sequences with a number of ones divisible by 15 could survive in both environments.

5 Acceptable Error Rates for DNA Tile-Based Evolution

While several experimental studies have shown that DNA tiles can process information through cooperative binding [11,15], it is also becoming clear that errors occur often during algorithmic assembly [15]. This is a concern because a low error rate is vital to the design of a self-replicator. If the error rate exceeds an error threshold [7], genetic meltdown occurs and sequences become totally random. In this section we describe how to decrease the error rate below any relevant error threshold.

Errors during assembly occur when a tile binds to a growing assembly by fewer than two bonds, an event called an unfavorable attachment. A mismatch error, an unfavorable attachment that only partially matches the adjacent tiles, causes an error in replication (Figure 5a). Additionally, in the absence of a pre-existing crystal, a series of unfavorable attachments occasionally produces a full-width crystal with a random sequence, an event called spontaneous nucleation.

Both these kinds of errors can be analyzed using a reversible model of DNA tile self-assembly based on the physics and chemistry of DNA hybridization [19]. Prior work on the robustness of algorithmic self-assembly in this model can be adapted in order to show that, at a moderate cost of tile set complexity and assembly speed, mismatch error rates can be made as small as is desired. "Proof-reading" tile sets implement the same logic of an original tile set but assemble more robustly, dramatically reducing mismatch error rates without significant slow-down [20,6,14]. The general idea of proofreading is to redundantly encode each element of sequence. When the proofreading method is applied to the zig-zag tile set (Figure 5b), correct tile additions are stabilized by additional tiles in the same block that encode the same sequence element, whereas several incorrect additions instead of just one are needed to propagate a sequence element incorrectly (Figure 5c). Error rates decrease exponentially as larger blocks of proofreading tiles are used [20].

Similar error correction techniques also exist for the prevention of spontaneous nucleation errors. Like other crystallization processes, the rate at which spontaneous nucleation of growing zig-zag assemblies occurs is dependent on the energy of the critical nucleus for growth. For zig-zag crystals, this critical nucleus is a small assembly that contains both a top and bottom boundary tile. By increasing the minimum width of an assembly that can contain both these tiles, it is possible to increase the energy of the critical nucleus. For example, the rate of spontaneous nucleation of the zig-zag tile set shown in Figure 3a decreases

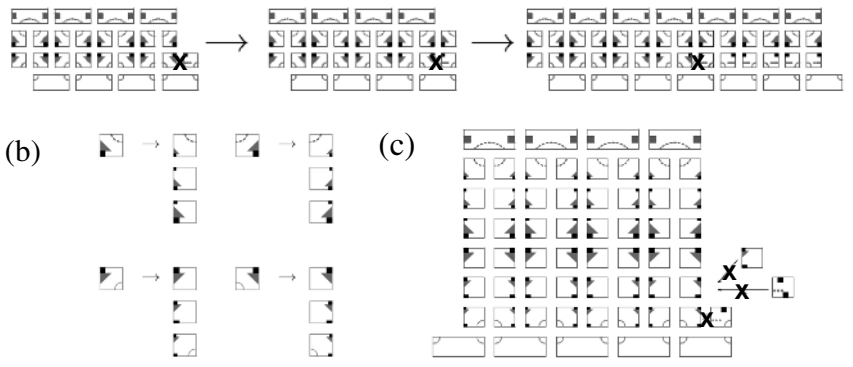


Fig. 5. Proofreading for zig-zag assembly. (a) Kinetic trapping is the major cause of mismatch errors in DNA tile assembly. When a tile attaches to an assembly on only one side, it forms a low energy bond and usually dissociates quickly. However, if another tile attaches to the assembly at an adjacent location before it can dissociate, the tile may be trapped. The mutated sequence will be copied to subsequent columns. **(b)** Zig-zag proofreading transformations of the four zig-zag middle tiles in Figures 2b. **(c)** Zig-zag assembly of the original sequences using the transformed tile set. When a single tile that produces an error attaches to the assembly, either the tile must fall off and be replaced by the correct tile, or further errors are necessary in order to continue growth.

exponentially with the width n [16]. We expect that the same qualitative result applies to the more complex tile sets described in this paper.

6 Conclusions

To study the physical principles of Darwinian evolution, we propose a physical system based on DNA crystals in which a combinatorial variety of genotypes can be faithfully replicated and a genotype can direct a behavior or other measurable parameter that can be subject to selection. DNA crystals are simple, containing no biological parts, and can be programmed to replicate an infinite variety of genotypes. The ability to program the interactions between tiles allows us to induce selection pressures which favor the growth of assemblies with interesting properties. Error correction techniques exist which can lower the replication error rate as much as is required to avoid genetic meltdown, at the cost of a small amount of additional complexity.

Acknowledgements. We would like to thank Bernie Yurke, Gerald Joyce, Andy Ellington, Graham Cairns-Smith, Paul Rothemund, Dave Zhang and David Soloveichik for helpful discussion on sequence amplification and evolution. The AFM image in the inset of Figure 1c was taken by Ho-Lin Chen. This research was partially supported by NSF awards #0093486 and #0432193.

References

1. Leonard M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, November 11, 1994.
2. Leonard M. Adleman, Jarkko Kari, Lila Kari, and Dustin Reishus. On the decidability of self-assembly of infinite ribbons. *Symposium on Foundations of Computer Science (FOCS)*, 43:530, 2002.
3. A. Graham Cairns-Smith. The origin of life and the nature of the primitive gene. *Journal of Theoretical Biology*, 10:53–88, 1966.
4. A. Graham Cairns-Smith. The chemistry of materials for artificial Darwinian systems. *International Revs. Phys. Chem.*, 7:209–250, 1988.
5. Nickolas Chelyapov, Yuriy Brun, Manoj Gopalkrishnan, Dustin Reishus, Bilal Shaw, and Leonard Adleman. DNA triangles and self-assembled hexagonal tilings. *J. Am. Chem. Soc.*, 126(43):13924–13925, 2004.
6. Ho-Lin Chen and Ashish Goel. Error free self-assembly using error prone tiles. In *DNA Computing 10*, Berlin Heidelberg, 2004. Springer-Verlag.
7. Manfred Eigen. Self-organization of matter and evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523, 1971.
8. Tsu-Ju Fu and Nadrian C. Seeman. DNA double-crossover molecules. *Biochemistry*, 32:3211–3220, 1993.
9. Gerald F. Joyce. Nonenzymatic template-directed synthesis of informational macromolecules. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 52, pages 41–51, 1987.
10. Thomas H. LaBean, Hao Yan, Jens Kopatsch, Furong Liu, Erik Winfree, John H. Reif, and Nadrian C. Seeman. Construction, analysis, ligation and self-assembly of DNA triple crossover complexes. *J. Am. Chem. Soc.*, 122(9):1848–1860, 2000.
11. Chengde Mao, Thomas H. LaBean, John H. Reif, and Nadrian C. Seeman. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature*, 407(6803):493–496, 2000.
12. Melanie Mitchell, Stephanie Forrest, and John H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proceedings of the First European Conference on Artificial Life*, 1992.
13. John H. Reif. Local parallel biomolecular computation. In Harvey Rubin and David Harlan Wood, editors, *DNA Based Computers III*, volume 48 of *DIMACS*, pages 217–254, Providence, RI, 1997. American Mathematical Society.
14. John H. Reif, Sudheer Sahu, and Peng Yin. Compact error-resilient computational DNA tiling assemblies. In *DNA Computing 10*, Berlin Heidelberg, 2004. Springer-Verlag.
15. Paul W. K. Rothemund, Nick Papadakis, and Erik Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLOS Biology*, 2:424–436, 2004.
16. Rebecca Schulman and Erik Winfree. Controlling nucleation rates in algorithmic self-assembly. In *DNA Computing 10*, Berlin Heidelberg, 2004. Springer-Verlag.
17. Erik Winfree. Self healing tile sets for algorithmic self-assembly. In preparation.
18. Erik Winfree. On the computational power of DNA annealing and ligation. In Richard J. Lipton and Eric B. Baum, editors, *DNA Based Computers*, volume 27 of *DIMACS*, pages 199–221, Providence, RI, 1996. American Mathematical Society.
19. Erik Winfree. Simulations of computing by self-assembly. Technical Report CS-TR:1998.22, Caltech, 1998.
20. Erik Winfree and Renat Bekbolatov. Proofreading tile sets: Error-correction for algorithmic self-assembly. In Junghuei Chen and John Reif, editors, *DNA Computing 9*, volume LNCS 2943, pages 126–144, Berlin Heidelberg, 2004. Springer-Verlag.
21. Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394:539–544, 1998.

All Else Being Equal Be Empowered

Alexander S. Klyubin, Daniel Polani, and Chrystopher L. Nehaniv

Adaptive Systems Research Group,
School of Computer Science, Faculty of Engineering and Information Sciences
University of Hertfordshire, College Lane, Hatfield Herts AL10 9AB, UK
{A.Kljubin, D.Polani, C.L.Nehaniv}@herts.ac.uk

Abstract. The classical approach to using utility functions suffers from the drawback of having to design and tweak the functions on a case by case basis. Inspired by examples from the animal kingdom, social sciences and games we propose *empowerment*, a rather universal function, defined as the information-theoretic capacity of an agent's actuation channel. The concept applies to any sensorimotoric apparatus. Empowerment as a measure reflects the properties of the apparatus as long as they are observable due to the coupling of sensors and actuators via the environment.

1 Introduction

A common approach to designing adaptive systems is to use utility functions which tell the system which situations to prefer and how to behave in general. Fitness functions used in evolutionary algorithms are similar in spirit. They specify directly or indirectly which genotypes are better.

Most utility functions and fitness functions are quite specific and a priori. They are designed for the particular system and task at hand and are thus not easily applicable in other situations. Each time the task and the properties of the system have to be translated into the "language" of the utility or fitness function. How does Nature address this problem? Is there a more general principle?

One common solution found in living organisms is homeostasis [1]. Organisms may be seen to maintain "essential variables", like body temperature, sugar levels, pH levels. Homeostasis provides organisms with a local gradient telling which actions to make or which states to seek. The mechanism itself is universal and quite simple, however the choice of variables and the methods of regulation is not. They are evolved and are specific to different phyla.

2 Empowerment

2.1 Motivation

Our central hypothesis is that there exist a *local* and *universal* utility function which may help individuals survive and hence speed up evolution by making the fitness landscape smoother. The function is local in the sense that it doesn't

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 744–753, 2005.

rely on an infinitely long history of past experience, does not require global knowledge about the world. The utility function is applicable to all species, hence, it should be universal. At the same time it should adapt to morphology and ecological niche. The utility function should be related to other biologically relevant quantities.

In the quest for the function one invariably notices certain traits reappear in different contexts over and over again. In animal kingdom we see the striving for domination and control. Humans and even states strive for money, power and control. In board games such as Reversi or Othello there is a concept of mobility, which is defined as the number of moves a player can make. Everything else being equal players should seek higher mobility.

The unifying theme of these and many other examples is the striving towards situations where *in the long term* one could do many different things *if one wanted to*, where one has more *control* or influence over the world. Predators with better sensors and actuators can hunt better. Having high status in a group of chimpanzees allows one more mating choice. Having a lot of money enables one to engage in more activities. One can choose from an array of options. However, if one doesn't know what to do, a good rule of thumb is to choose actions leading to higher status, more power, money and control. We will now apply this idea to "embodied" agents.

2.2 The Concept of Empowerment

In his work on ecological approach to visual perception [2] Gibson proposed that animals and humans do not normally view the world in terms of geometrical space, independent arrow of time, and Newtonian mechanics. Instead, he argued, the natural description is in terms of what one can perceive and do. Thus, different places in the world are characterized by what they afford one to perceive and do.

This perspective is agent-centric. The concept of "the environment" is a by-product of the interplay between the agent's sensors and actuators. In this spirit we base our utility function solely on the sensors and actuators, without the need to refer to the "outside" of the agent.

We propose *empowerment*, a quite general utility function, which only relies on the properties of "embodiment", the coupling of sensors and actuators via the environment. Empowerment is *the perceived amount of influence or control* the agent has over world. For example, if the agent can make one hundred different actions but the result, *as perceived by the agent*, is always the same, the agent has no control over the world whatsoever. If, on the other hand, the agent can reliably force the world into two states distinguishable by the agent, it has two options and thus two futures to choose from. Empowerment can be seen as the agent's *potential* to change the world, that is, how much the agent could do in principle. This is in general different from the *actual* change the agent inflicts.

In the section 2.4 we will quantify empowerment using Information Theory [3]. Briefly, *empowerment is defined as the capacity of the actuation channel*

of the agent. The main advantage of using Information Theory for defining empowerment is that the measure is universal in the sense that it does not depend on the task or on the "meaning" of various actions or states.

2.3 The Communication Problem

Here we provide a brief overview of the classical communication problem from Information Theory and define channel capacity for a discrete memoryless channel. For an in depth treatment we refer the reader to [3,4].

There is a sender and a receiver. The sender transmits a signal, denoted by a random variable X, to the receiver, who receives a potentially different signal, denoted by a random variable Y. The communication channel between the sender and the receiver defines how transmitted signals correspond to received signals. In the case of discrete signals the channel can be described by a conditional probability distribution $p(y|x)$.

Given a probability distribution over the transmitted signal, *mutual information* is defined as the amount of information, measured in *bits*, the received signal on the average contains about the transmitted signal. Mutual information can be expressed as a function of the probability distribution over the transmitted signal $p(x)$ and the distribution characterizing the channel $p(y|x)$:

$$I(X;Y) = \sum_{\mathcal{X},\mathcal{Y}} p(y|x)p(x) \log_2 \frac{p(y|x)}{\sum_{\mathcal{X}} p(y|x)p(x)} \ . \tag{1}$$

Channel capacity is defined as the maximum mutual information for the channel over all possible distributions of the transmitted signal:

$$C = \max_{p(x)} I(X;Y) \ . \tag{2}$$

Channel capacity is the maximum amount of information the received signal can contain about the transmitted signal. Thus, mutual information is a function of $p(x)$ and $p(y|x)$, whereas channel capacity is a function of the channel $p(y|x)$ only. Another important difference is that mutual information is symmetric in X and Y and is thus acausal, whereas channel capacity requires complete control over X and is thus asymmetric and causal (cf. [5]).

There exist efficient algorithms to calculate the capacity of an arbitrary discrete channel, for example, the iterative algorithm by Blahut [6].

2.4 Definition of Empowerment

For the sake of simplicity of the argument, let us assume a memoryless agent in a world. Following the information-theoretic approach to modeling perception-action loops described in [7,8] we can split the whole system into the agent's

sensor, the agent's actuator and the rest of the system[1] including the environment. The states of sensor, actuator and the rest of the system at different time steps are modeled as random variables (S, A, and R respectively). The perception-action loop connecting these variables is unrolled in time. The pattern of dependencies between these variables can be visualized as a Bayesian network (Fig. 1).

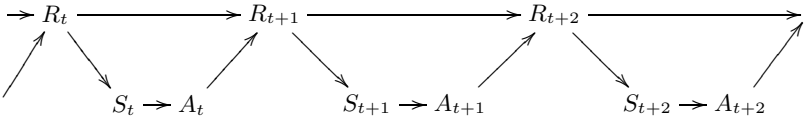


Fig. 1. The perception-action loop as a Bayesian network. S – sensor, A – actuator, R – rest of the system. R is included to formally account for the effects of the actuation on the future sensoric input. R is the state of the actuation channel.

Previously we colloquially defined empowerment as the amount of influence or control the agent has over the world as perceived by the agent. We will now quantify the amount of influence as the amount of Shannon information[2] the agent could "imprint onto" or "inject into" the sensor. Any such information will have to pass through the agent's actuator.

When will the "injected" information reappear in the agent's sensors? In principle, the information could be "smeared" in time. For the sake of simplicity in this paper will be using a special case of empowerment: *n-step sensor empowerment*. Assuming that the agent is allowed to perform any actions for n time steps, what is the *maximum* amount of information it can "inject" into the momentary reading of its sensor after these n time steps (Fig. 2)? The more of the information can be made to appear in the sensor, the more control or influence the agent has over its sensor.

We view the problem as the classical problem of communication from Information Theory [3] as described in Sec. 2.3. We need to measure the maximum amount of information the agent *could* "inject" or transmit into its sensor by performing a sequence of actions of length n. This is precisely the capacity of the channel between the sequence of actions and sensoric input n time steps later.

Let us denote the sequence of n actions taken, starting at step t, as a random variable $A_t^n = (A_t, A_{t+1}, \ldots, A_{t+n-1})$. Let us denote the state of the sensor

[1] We include the rest of the system, denoted by R, only to account for the effects of actuation on the future sensoric input. R is the state or memory of the actuation channel. For the problem of channel with side information it is established [4] that knowing the state of the channel may increase its capacity. Thus, in addition to actuator, sensor and the rest of the system it is useful to define *context*, a random variable approximating the state of the actuation channel in a compact form (cf. Information Bottleneck [9], ϵ-machines [10,11]). However, we omit this more general treatment from the present discussion.

[2] The word "information" is always used strictly in the Shannon sense in this paper.

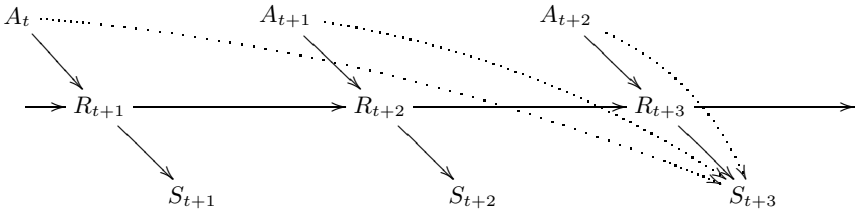


Fig. 2. 3-step sensor empowerment. Actions are independent of system's state (agent with "free will"). The communication channel goes from actions (A_t, A_{t+1}, A_{t+2}) to sensor S_{t+3}.

n time steps later by a random variable S_{t+n}. We now view A_t^n as the transmitted signal and S_{t+n} as the received signal. The system's dynamics induce a conditional probability distribution $p(s_{t+n}|a_t^n)$ between the sequence of actions A_t^n and the state of sensor after n time steps S_{t+n}. This conditional distribution describes the communication channel we need.

We define *empowerment* as the channel capacity of the agent's actuation channel terminating at the sensor (see Eq. 1 and Eq. 2):

$$\mathfrak{E} = C = \max_{p(a_t^n)} \sum_{\mathcal{A}^n, \mathcal{S}} p(s_{t+n}|a_t^n) p(a_t^n) \log_2 \frac{p(s_{t+n}|a_t^n)}{\sum_{\mathcal{A}^n} p(s_{t+n}|a_t^n) p(a_t^n)} \ . \tag{3}$$

Empowerment is measured in *bits*. It is zero when the agent has no control over what it is sensing, and it is higher the more perceivable control or influence the agent has. Empowerment can also be interpreted as the amount of information the agent could potentially "inject" [8] into the environment via its actuator and later capture via its sensor.

The maximizing distributions over the sequences of actions can be interpreted as distributions of actions the agent should follow in order to inject the maximum amount of information into its sensors after n time steps.

The conditional probability distribution $p(s_{t+n}|a_t^n)$ may induce equivalence classes over the set of sequences of actions. For example, if the various sequences of actions produce only two different outcomes in terms of the resulting probability distribution of sensoric input $p(s_{t+n})$ then the agent may view all the sequences of actions just in terms of two meta-actions corresponding to the two different distributions over the resulting sensoric input.

3 Experiments

In this section we present two experiments to illustrate the concept of empowerment. The first experiment demonstrates how an agent's empowerment looks in a grid world and how it changes when a box is introduced. The second experiment illustrates empowerment of an agent in a maze.

3.1 Box Pushing

Consider a two-dimensional infinite square grid world. An agent can move in the world one step at a time into one of the four adjacent cells. The actuator can perform five actions: go left, right, forward, back, and do nothing. For the sake of simplicity, let's assume that the agent has a sensor which reports the agent's absolute position in the world. What is this agent's n-step empowerment?

For this scenario the n-step empowerment turns out to be the logarithm of the number of different cells the agent can reach in n time steps: $\log_2(2n^2 + 2n + 1)$. This is $\log_2 5$ for 1 step, $\log_2 13$ for 2 steps, and so forth. The empowerment does not depend on where the agent starts with the sequence of actions (Fig. 3, b).

We now add a box occupying a single cell. The agent's sensor, in addition to the agent's position, now also captures the absolute position of the box. Let us assume that the box cannot be moved by the agent and thus remains stationary. If the agent tries to move into the cell occupied by the box the agent remains where it was. In this case the agent's empowerment is lower the closer the agent is to the box (Fig. 3, c). This can be explained by the fact that the box blocks some paths, and as a result it may render unreachable some of the previously reachable cells. Empowerment is high in the box because from there the agent can reach the maximum number of cells including the one occupied by the box.

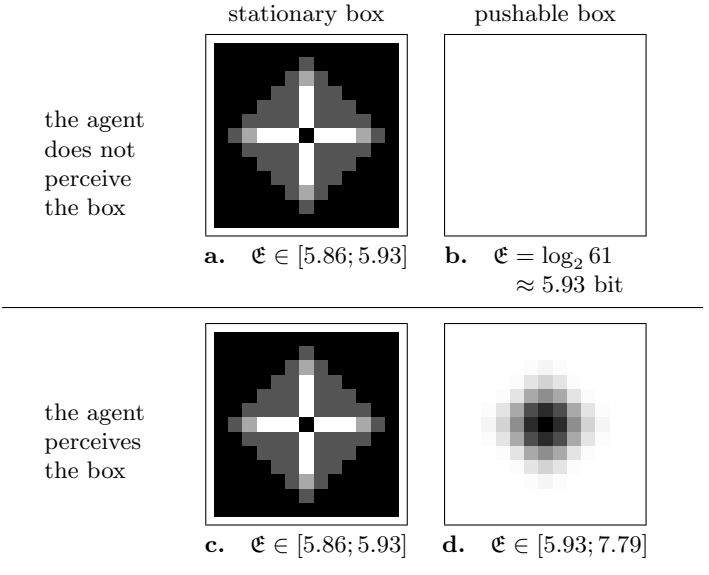


Fig. 3. 5-step empowerment field over the grid. The field is centered at the box. Because empowerment in cells further than 5 cells away from the box is always $\log_2(61) \approx 5.93$ bits, only the 13×13 cells central part of the field is shown. Cells are colored according to scaled empowerment of the agent in the cell. *Darker* color means *higher* empowerment. Maps are scaled independently of each other. Corresponding ranges of empowerment are provided below the maps. Note that the ranges are different in size.

Let us now assume that the box can be pushed by the agent. If the agent tries to move into the cell occupied by the box, it succeeds and the box is pushed in the direction of the agent's move. Empowerment is now more complex than just the number of cells reachable by the agent, because it also includes the position of the box. In this scenario the agent's empowerment in a given cell is the binary logarithm of the number of unique combinations of the agent's and the box's final positions achievable from a given cell. The agent's empowerment is higher the closer the agent is to the box (Fig. 3, d). The number of cells the agent can reach in n time steps is still the same as for the case without the box. However, some paths leading to same cells after n steps can now be differentiated by different positions of the box, because it was pushed differently. Thus, because the position of the box is observable and controllable by the agent, it can be viewed as an extra reservoir for empowerment.

It is also interesting to see what happens if the agent doesn't perceive the box, that is when the sensor captures only the agent's position. In the case of the stationary box, the empowerment field does not change (Fig. 3, a is identical to Fig. 3, c). This is because the position of the box never changes. Excluding it out from the sensor thus cannot decrease the amount of control over the sensor. With a stationary box, a sensor for the box's absolute position is useless. Having no sensor for the box, just by noticing the change in the conditional probability distribution $p(s_{t+n}|a_t^n)$ describing the actuation channel the agent could infer that something changed in the world (no box $\rightarrow$ stationary box).

In the case of the pushable box leaving out the position of the box from the sensor results in the completely flat empowerment field over the grid (Fig. 3, b), exactly as in the initial setup without the box. This is because the movement of the agent and hence its position is not influenced by the box at all. Thus, if the agent doesn't see the box, it cannot perceive it even indirectly.

To summarize, empowerment as a general utility function in this scenario translates to a simple measure of reachability for simple cases (no box, stationary box). Furthermore, it reacts reasonably to changes in the dynamics of the world, which do not need to be explicitly encoded into empowerment. We believe that empowerment discovers intuitively interesting places in the world.

3.2 Maze

Consider a two-dimensional square grid world. Similar to the previous scenario an agent moves in the world one step at a time into one of the four adjacent cells. Some cells have walls between them preventing the agent from moving. A maze is formed using the walls (Fig. 4). The agent has a sensor which captures the agent's global position.

We measure the n-step empowerment of the agent. Similar to the previous scenario, because of deterministic actuation and the nature of the sensor, empowerment is the logarithm of the number of the cells reachable in n moves. Empowerment maps for several time horizons are shown on Fig. 5.

A natural measure for navigation in mazes is the average shortest path from a given cell to any other cell. To navigate through any place in the maze fastest

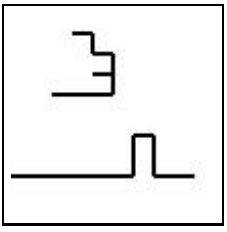


Fig. 4. A 10×10 maze. Walls between cells are shown in black.

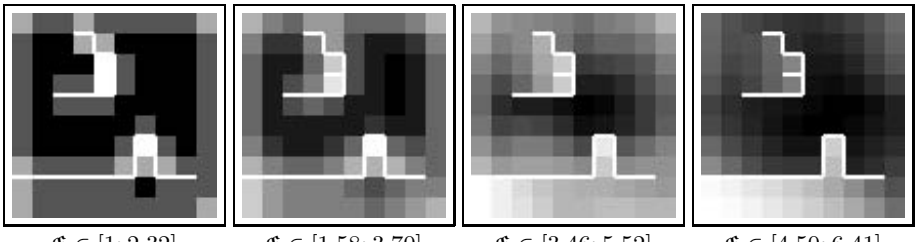


Fig. 5. 1-, 2-, 5-, and 10-step empowerment field over a 10×10 maze (left to right). Walls are shown in white. Cells are colored according to empowerment. *Darker* color corresponds to *higher* empowerment in the cell. Maps are scaled independently of each other. Corresponding empowerment ranges are shown below each map.

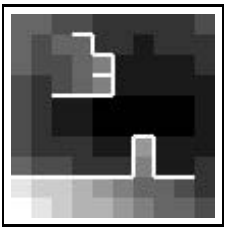


Fig. 6. Map of average shortest distance to other cells. *Darker* color corresponds to *lower* average distance.

one would want to start in a cell with lowest average distance to any other cell. The map of average shortest distances is shown on Fig. 6. It is similar to the map obtained using empowerment with several time steps. In fact, empowerment and average shortest path are roughly anti-correlated (See Fig. 7). However, the two types of maps need not coincide. For instance, if the task were to avoid a predator, the average distance map would not be of much help. However, n-step empowerment with straightforward modifications[3] would implicitly include the effects of the predator into the picture.

[3] A natural way to make the presence of the predator "known" to empowerment is to assume that once the agent is dead, for example, eaten by the predator, all actions have the same effect. As a result, empowerment drops to zero.

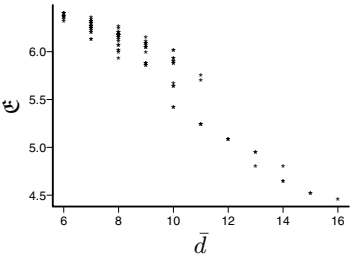


Fig. 7. 10-step empowerment of cells (vertical) vs. the average distance to other cells (horizontal)

4 Discussion and Conclusions

In the search for a general principle for adaptive behavior we have introduced empowerment, a natural and universal quantity derived from an agent's "embodiment", the relation between its sensors and actuators induced by the environment. Empowerment is defined for any agent, regardless of its particular sensorimotor apparatus and the environment, as the information-theoretic capacity of the actuation channel. Empowerment maximization, as a utility or fitness function, can be colloquially summarized as "everything else being equal, keep your options open."

We have shown two simple examples where the empowerment measure captures features of the world which have not and need not be specially encoded. For example, in the box pushing scenario, if the box is pushable the agent is more empowered the closer it is to the box, if the box is not pushable the agent is, vice versa, less empowered the closer it is to the box.

The presence of the box need not be "encoded" into empowerment at all. In both cases empowerment was calculated identically, the sensor and the actuator over which empowerment was measured remained unchanged. It was the dynamics of the world that changed, and empowerment generalized naturally to capture the change. The result was different depending on whether the box was pushable or not.

In the example with walking in a maze, empowerment is anti-correlated with the average shortest distance from a cell to any other cell. However, these two measures will cease to coincide, if, for example, a predator were introduced.

Our central hypothesis is that similar to the two simple examples, where empowerment in most cases was related to the number of reachable cells, empowerment maximization may translate into simpler measures and interpretations, like homeostasis, phototaxis, avoidance, etc.

Empowerment is useful for a number of reasons. Firstly, it is defined universally and independently of a particular agent or its environment. Secondly, it has a simple interpretation – it tells the agent to seek situations where it has control over the world and can perceive the fact. Thirdly, if the agent were to estimate empowerment on-board, it would know what actions lead to what situ-

ations in the future – this knowledge could be used for standard planning. Last but not least, empowerment can be calculated on-board in an agent-centric way or externally, as, for example, a fitness function in evolutionary search. In the latter case the agent need not know anything about empowerment – it would just behave as though it maximizes empowerment.

References

1. Ashby, W.R.: An Introduction to Cybernetics. Chapman & Hall Ltd. (1956)
2. Gibson, J.J.: The Ecological Approach to Visual Perception. Houghton Mifflin Company, Boston (1979)
3. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal **27** (1948) 379–423
4. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons, Inc. (1991)
5. Pearl, J.: Causality: Models, Reasoning, and Inference. Cambridge University Press (2001)
6. Blahut, R.: Computation of channel capacity and rate distortion functions. IEEE Transactions on Information Theory **18**(4) (1972) 460–473
7. Touchette, H., Lloyd, S.: Information-theoretic approach to the study of control systems. Physica A **331**(1–2) (2004) 140–172
8. Klyubin, A.S., Polani, D., Nehaniv, C.L.: Tracking information flow through the environment: Simple cases of stigmergy. In Pollack, J., Bedau, M., Husbands, P., Ikegami, T., Watson, R.A., eds.: Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, The MIT Press (2004) 563–568
9. Tishby, N., Pereira, F.C., Bialek, W.: The information bottleneck method. In: Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing. (1999) 368–377
10. Crutchfield, J.P., Young, K.: Inferring statistical complexity. Physical Review Letters **63**(2) (1989) 105–108
11. Shalizi, C.R., Crutchfield, J.P.: Information bottlenecks, causal states, and statistical relavance bases: How to represent relevant information in memoryless transduction. Advances in Complex Systems **5**(1) (2002) 91–95

Artificial Homeostatic System: A Novel Approach

Patrícia Vargas[1], Renan Moioli[1], Leandro N. de Castro[2], Jon Timmis[3], Mark Neal[4], and Fernando J. Von Zuben[1]

[1] DCA/FEEC/Unicamp - Brazil
{pvargas, moioli, vonzuben}@dca.fee.unicamp.br
[2] Unisantos - Brazil
lnunes@unisantos.br
[3] University of Kent - UK
J.Timmis@kent.ac.uk
[4] University of Wales - UK
mjn@aber.ac.uk

Abstract. Many researchers are developing frameworks inspired by natural, especially biological, systems to solve complex real-world problems. This work extends previous work in the field of biologically inspired computing, proposing an artificial endocrine system for autonomous robot navigation. Having intrinsic self-organizing behaviour, the novel artificial endocrine system can be applied to a wide range of problems, particularly those that involve decision making under changing environmental conditions, such as autonomous robot navigation. This work draws on "embodied cognitive science", including the study of intelligence, adaptivity, homeostasis, and the dynamic aspects of cognition, in order to help lay down fundamental principles and techniques for a novel approach to more biologically plausible artificial homeostatic systems. Results from using the artificial endocrine system to control a simulated robot are presented.

1 Introduction

In previous work, Timmis and Neal [22] presented a model for an artificial endocrine-system (AES) as a module of a broader conceptual framework including artificial neural networks (ANN) and artificial immune systems (AIS), with the ultimate goal of developing an artificial homeostatic system (AHS). The AHS (e.g., a mobile robot) is capable of autonomously interacting with an unknown and changing environment while maintaining its internal state (e.g., energy level and integrity) and optimizing some objectives. It is now believed in biology that there is an interconnection and dependence among the immune, nervous and endocrine systems, which is fundamental for cognition, maintenance of the internal state of an organism (called homeostasis), immune-regulation and host defenses [1]. The present work concentrates on the neuro-endocrine interactions and mechanisms in order to create a more biologically plausible artificial homeostatic system. The approach borrows some ideas from "embodied intelligence", introduced by Rodney Brooks [2][3][4] to synthesize a cognitive system based on coupled dynamics and nonstationary mappings.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 754 – 764, 2005.

The paper is organized as follows: Section 2 gives a brief description of the fundamentals of the nervous and endocrine systems, followed by some key interactions between the two systems and the biological mechanisms that motivated this work. Section 3 revisits some aspects of the embodied cognitive science related to robot autonomous navigation. In Section 4 the previous model of the artificial homeostatic system is presented, and Section 5 introduces the novel artificial homeostatic system. Some preliminary simulations are presented in Section 6. Discussions and future work compose Section 7.

2 Nervous and Endocrine Systems Interactions

The nervous system (NS) is primarily responsible for the reception of stimuli, by detecting changes in the internal and external environments, and for processing and transmitting the nerve impulses as appropriate responses to those changes [14].

The endocrine system can be viewed as a system of glands [20] that works with the nervous system in controlling the activity of internal organs and in coordinating the long-range response to external stimuli. The main roles of the endocrine system are to assist the maintenance of homeostasis, growth, differentiation, metabolism, reproduction, and to help the organism to cope with stress [14][15]. All these tasks are accomplished by the hormones, which are chemical substances produced, stored and secreted by the components of the endocrine system, including a group of glands, specialized cells, body tissues and organs.

One of the most important neuro-endocrine interactions happens within the hypothalamus and the pituitary gland (or hypophysis). The hypothalamus is a region in the brain beneath the thalamus. It consists of many aggregations of nerve cells and its main function is to control release of pituitary hormones. The hypothalamus is responsible for the integration of many basic behavioural patterns, involving the correlation of neural and endocrine functions. Its neurons are also affected by a variety of hormones and other circulating chemicals [10]. In fact, due to this interaction, in the last three decades the hypothalamus has been often referred to as the "endocrine hypothalamus" [9][20]. The pituitary gland is located at the base of the brain. The hypothalamus controls the release of hormones from the pituitary that will in turn control other target organs, and also other target endocrine glands.

This hypothalamus-pituitary interaction is controlled by feedback mechanisms. There exists a positive feedback when the production and release of hormones is excitatory. Nonetheless, normally these physiological functions are under a negative feedback mechanism regulation, in which the hormone production and release is inhibitory [10]. Feedback is carried out by three mechanisms: a sensor that will sense the controlled variables under supervision, a reference point, and an error signal. Release of some hypothalamic hormones is ruled by external and internal neural inputs (short loops), and also by long feedback loops involving remote organs and external metabolic processes [9].

The important point is that the release of hormones influences nervous activity (thus cognition, motor control, etc.), whilst nervous activity influences endocrine function (thus growth, metabolism, etc.), in a semi-closed control loop: internal

processes are allowed to regulate themselves whilst external environmental factors can also regulate and control the system. In the novel homeostatic system to be developed here, an artificial endocrine system will be adopted to aid an artificial neural network in the process of robot autonomous navigation.

3 Robot Autonomous Navigation – A Cognitive Challenge

Attempts to understand the mind and its operation have been ongoing since the ancient Greeks philosophers [21]. This study has lead to the development of cognitive science. Simply put, cognitive science is the interdisciplinary study of mind and intelligence [19]. This in turn has lead to the appearance of a novel field of research at the end of the 1980's, called "embodied cognitive science", also known as "behaviour-based robotics". The new term was based on the ideas of "embodied intelligence" coined by Rodney Brooks [2][3][4]. For Brooks, the only way to understand intelligence is by giving it a body, i.e. by turning it into an embodied system it will be possible to focus upon the interaction between the embodied system and the real world, therefore releasing it from the human interpreter.

These ideas are best understood when potential applications for intelligent robots are considered. Sea and submarine prospecting, space exploration, discovery of mines, firefighting, military assistance, and search and rescue services are among the plethora of challenges that might be faced by an autonomous mobile robot. There are many ways of conducting artificial agent experiments productively and systematically, among which there are experiments designed for simulated and real robotic agents. The merits of simulation versus physical embodiment are still under discussion (see Pfeifer and Scheier [19]). In our work both approaches are adopted. There is a real robotic agent, the Khepera II® Robot, and a simulated robot agent, using the WSU Khepera Simulator [18].

Among the methods for tackling robot autonomous navigation problems, it is worth highlighting evolutionary approaches [6][7][8][17]. Evolutionary theory proposes that the brain has evolved to control behaviour in order to ensure our survival [19]. Additionally, it is agreed that intelligence manifests itself in behaviour and thus we must understand behaviour before we can completely understand intelligence and therefore create embodied intelligence. Toward this goal, another extremely important concept, which can be considered behaviour–based, is adaptivity, i.e. the ability to adapt to a continuously changing and unpredictable environment. In fact, there is a direct relation between intelligence and adaptivity [19]. During adaptation, some variables need to be kept within certain pre-determined bounds, either by evolutionary changes, physiological reactions, sensory adjustment, or by simply learning novel behaviours. This definition of adaptivity has to do with the concept of homeostasis, a term first coined by Ashby in 1960 [19]. Within the limits controlled by homeostatic processes, the organism or the artificial agent can function and stay alive in a "viability zone". While trying to design artificially homeostatic (and self-organizing) systems, this work proposes a novel biologically plausible artificial homeostatic system based on previous work for autonomous navigation [22].

4 Artificial Homeostatic System – Previous Work

In previous work, Timmis and Neal [22] presented a model for an artificial endocrine system (AES), which would be part of a broader conceptual framework including artificial neural networks (ANN) and artificial immune systems (AIS) (Figure 1). The system was based on the mammalian body and its mechanisms for the maintenance of homeostasis, where the authors have chosen an intermediate level of granularity.

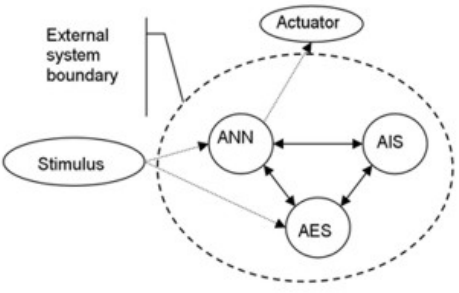


Fig. 1. Artificial homeostatic system overview (adapted from Timmis and Neal [22])

The AES was described as a system that employed controlling hormones. In the present paper, only the AES and ANN interactions will be discussed (the entire homeostatic system is discussed in greater detail in Timmis & Neal [22]).

The ANN proposed uses a standard error backpropagation-learning algorithm to train a multi-layer perceptron (MLP) neural network [11]. Initially there is no explicit interaction between the ANN and the AES. The AES provides a medium-term regulatory control mechanism for the behaviour of the system. It consists of gland cells that secrete hormones stored using a pool mechanism in response to external stimuli:

$$r_g = \alpha_g \sum_{i=0}^{nx} x_i \tag{1}$$

where r_g is the quantity of hormone released by gland g; α_g is the rate at which hormones are released by gland g; x_i is the i-th input to gland g; and n_x is the number of inputs to gland g.

Given that $c_g(t)$ is the hormone concentration in gland g at a time t, then the variation in concentration obeys (where $\beta < 1$ is a decay constant):

$$c_g(t+1) = c_g(t) \times \beta \tag{2}$$

Gland cells secrete and record the concentration of hormones present in the system and use it to moderate the strength of reaction. Each gland cell secretes a specific hormone, represented by a simple bit-string. The hormone levels would affect the input weights in the ANN, i.e. the recorded hormone level would affect each input weight on a particular neuron as follows:

$$u = \sum_{i=0}^{nx} w_i x_i \prod_{j=0}^{ng} C_j S_{ij} M_{ij} \tag{3}$$

where u is the internal activation of the neuron, w_i is the weight for the i-th input x_i; n is the number of weights; n_g is the number of glands in the system; C_j is the concentration of hormone j; S_{ij} is the sensitivity of the connection from receptor i to hormone j; and M_{ij} is the match between receptor i and hormone j.

5 Artificial Homeostatic System: A Novel Approach

Towards the goal of designing a more biologically plausible model and a system with a greater potential to promote artificial homeostasis, we focused on mimicking some mechanisms of the endocrine system used to control the concentration level of hormones within the artificial agent (a simulated robot in our study). A novel artificial endocrine system is composed of three main modules: hormone level repository (HL), hormone production controller (HPC), and endocrine gland (G) (Figure 2). The hormone level repository has a record of the level of hormone in the organism; the hormone production controller is responsible for controlling the production of hormones in response to variations in the internal state of the organism and external stimulation; and the endocrine gland receives inputs from the HPC, being responsible for producing and secreting hormones when required.

Note that, in such a system, any variation in the internal and external states may promote or suppress the activity of the nervous (ANN) and endocrine (AES) systems. For instance, the variation of the internal state of the organism as a result of hormone production may act as a feedback mechanism to the hormone production itself, resulting in the release of inhibitory hormones or in the cessation of hormone production.

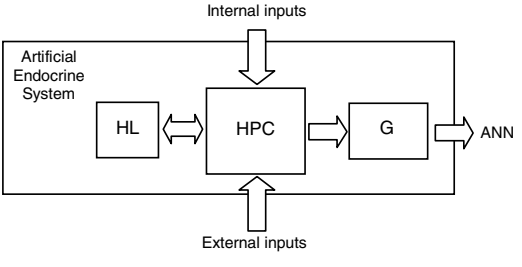


Fig. 2. The main components of the new AES and their interaction with the environment

The system dynamics is founded on some of the main biological mechanisms of homeostasis, particularly positive and negative feedback mechanisms of the endocrine system. The HPC module sends excitatory signals, which work as a positive feedback to the gland G, which in turn starts to produce and release hormone (without implementing the pool mechanism previously proposed [22]), thus increasing the hormone level. The level of hormone will in turn alter the internal state by driving neural network actions upon the environment. By sensing inhibitory signals that promote

negative feedback from the internal state, the HPC module ceases the production of excitatory signals (positive feedback) until once again it senses specific changes in the internal state.

5.1 System Dynamics

The internal state (IS) of the system described in this paper is modeling a seeking behaviour, i.e. it drops to zero or null state when the robot reaches the target. By drawing an analogy to human drives and desires, the target can be governed by many types of possible behaviors, such as, desire to eat: target = food; desire to charge: target = base; desire to wander: target = no collision.

The internal state of the artificial agent will depend on the level of external stimulus (ES) and also on the hormone level (HL) present within the artificial organism at instant t. If the ES and HL are above certain pre-determined thresholds (λ and ω, respectively), then the internal state is equal to zero; that is, the level of "desire" falls down to zero. This happens because there are enough external stimuli present and there are enough hormones to trigger the behaviour. Otherwise, the internal state will increase at a pre-determined rate β until it reaches a pre-defined maximum level Max(IS):

$$\begin{aligned} &\text{If } (\mathrm{ES} \geq \lambda) \text{ and } (\mathrm{HL} \geq \omega) \text{ then } \mathrm{IS} = 0 \\ &\text{else } \mathrm{IS}(t+1) = \mathrm{IS}(t) + \beta(\mathrm{Max}(\mathrm{IS}) - \mathrm{IS}(t)) \end{aligned} \quad \text{(Rule 1)}$$

The external stimulus (ES) depends on the proximity of the artificial agent to the targets. Analogous to the human body sensitivity to external stimuli, this distance can be sensed by the artificial agent using its sensor inputs. This information will be made available not only to the artificial neural network but also to the artificial endocrine system.

The hormone production controller (HPC) effectively "selects" in a completely sub-symbolic way the behaviour to be exhibited by the robot. The HPC mimics the hypothalamus and thus senses both the external environment and the internal state, and thus triggers the hormone production and release by the gland G. This triggering may cause the hormone concentration level to increase and therefore to stimulate its target cells (the neurons) to perform a certain task. While this task is not accomplished, the HPC will keep on producing and releasing hormones, thus maintaining a suitable hormone level. When the task is accomplished, i.e. when the HPC receives a negative feedback signal, it ceases the production and release of hormone. Based on this mechanism, Rule 2 synthesizes the control of hormone production and release:

$$\begin{aligned} &\text{If } \mathrm{IS} \geq \theta \quad \text{then} \quad \mathrm{HP}(t+1) = (100 - \%\mathrm{ES}) \times \alpha(\mathrm{Max}(\mathrm{HL}) - \mathrm{HL}(t)) \\ &\text{else} \quad \mathrm{HP} = 0 \end{aligned} \quad \text{(Rule 2)}$$

where θ is the target threshold of the internal state IS; HP is the hormone production; ES is the external stimulus; α is the scaling factor; HL is the hormone level; and t is the time index.

If the internal state IS is greater than or equal to a target threshold θ, then hormone will be produced at a rate that will depend upon the level of the external stimulus received and the level of hormone already present within the artificial organism.

Otherwise, if the internal state IS is less than a target threshold θ, then hormone production will cease.

The hormone level represents the amount of hormone stimulating the neural network (ANN). The hormone level will undergo a constant updating in its value due to its internal half-life measure [8] and the amount of hormone produced:

$$\mathrm{HL}(t+1) = \mathrm{HL}(t) \times e^{-1/T} + \mathrm{HP}(t) \tag{4}$$

where T is the hormone half-life.

6 Preliminary Experiments

In order to better clarify and assess the performance of the new AHS three preliminary experiments were conducted. Experiments I and II were designed for a simulated robotic agent using the WSU Khepera Robot Simulator [18] (Figure 3) and Experiment III was designed for a real robotic agent using the Khepera II Robot® [13]. The rationale behind these experiments is twofold: first, to show the systems' adaptivity through its ability to cope with internal and external changes; and, second, to confirm the systems' ability to adapt to a dynamical environment, by presenting the phenomenon of biological cyclic behaviour synchronized with the amount of resources available via homeostatic control.

6.1 Experiments with a Simulated Robotic Agent

The simulated robot first learned two separate tasks: to avoid collision and to detect a light source (here associated with a food source). Both tasks were learned via a standard error backpropagation-learning algorithm used to train two separate multilayer perceptron (MLP) neural networks [11]. The input-output training data is composed of samples from a diverse set of relevant navigation conditions. After training, the robot was introduced into the arena with walls (obstacles) and a light source (food source) in the middle (Figure 3).

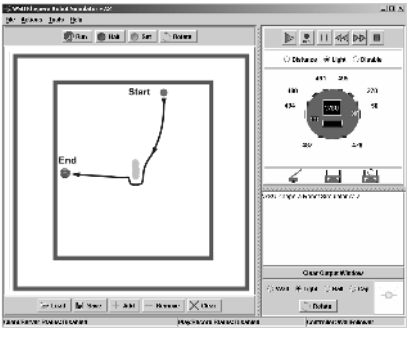
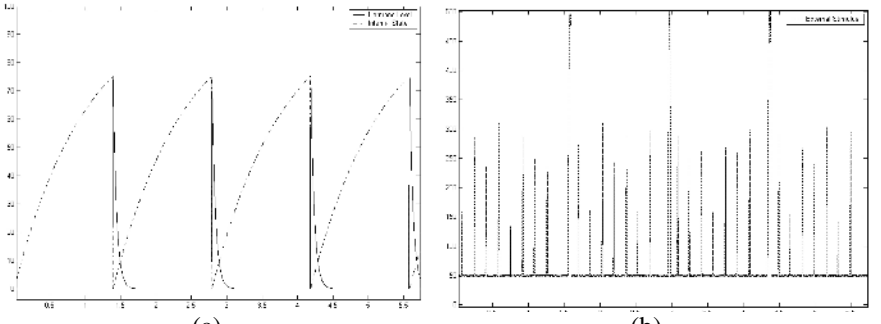


Fig. 3. Experiment I: Sequence of steps performed by the robot. The robot leaves the top right corner moving towards the light source (in the middle of the arena).

In Experiment I, once the robot begins to navigate, its behaviour was solely controlled by the AES, which was designed to manage its internal state "desire to eat" (here, *eat* means *recharge)* employing the endocrine mechanisms described previously. The robot's light sensor values ranged from 50 to 500 depicting the presence of external stimulus (high values mean lack of food). Note that the robot reaches the target, fulfils its "desire to eat" and then moves away from the target towards the left wall. All other values were printed from the system behaviour/reaction to this information. Figure 4a shows the hormone and the internal state levels and Figure 4b shows the external stimuli during 55,000 iterations or navigation steps. The hormone and internal state values ranged from 0 to 100 units at most. The internal state depicted in Figure 4a refers to the need of energy (desire to eat). The highest level of hormone production added to the proximity to the target caused the robot to fulfil its "desire to eat" four times during the simulation. This confirms the influence of the hormone level over the robot's autonomous behaviour. The parameter values adopted in this simulation were: $\beta = 0.0001$; $\alpha = 1.8$; $T = 500.0$; $\lambda = 150$; $\omega = 90.0$; and $\theta = 75.0$, and were defined empirically.



(a) (b)

Fig. 4. Experiment I: (a) Hormone and internal state levels.(b) Output of the light sensor along navigation.

Experiment II explored a scenario where the robot actions were influenced by two external stimuli, the distance to the target and a varying, cyclical availability of food, meaning that the higher the food quantity (FQ), the greater the amount of food available. This experiment was divided into two distinct simulations: A and B.

In Simulation A, the FQ was created based on a smooth sine curve function and its value had an influence over the parameter of the internal state (Rule 1), providing a self-adaptive behaviour for the robot in a way that it tends to eat only when the FQ value is higher than 50 (in a 0 to 100 scale) (Figure 5a). This adaptability accounts for the cyclic behaviour observed when the robot's drive to eat synchronized with the availability of food.

In Simulation B, the FQ was created based on a stepwise sine curve function (to facilitate a future simulation designed for a real robotic agent) and its value had also an influence over the β parameter of the robot's internal state (Figure 5b). For instance, the FQ could be implemented by an automatic discrete electronic apparatus for

the control and adjustment of the light source blinking speed. The faster the blinking speed, the greater the quantity of food available.

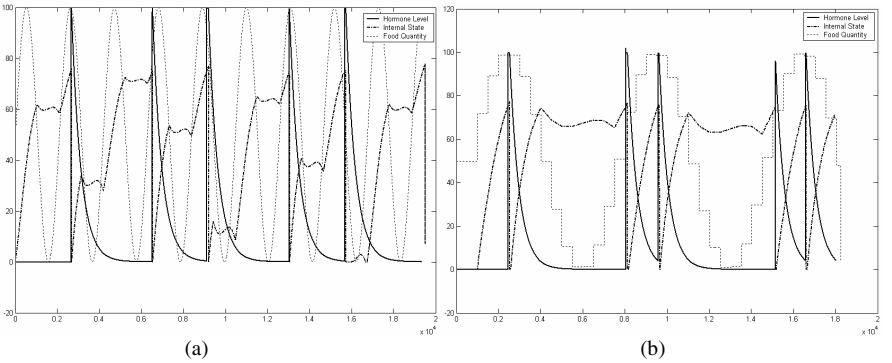


Fig. 5. (a)Experiment IIA: synchronous behaviour for a sinusoidal stimulus (b) Experiment IIB: synchronous behaviour for a cyclical stepwise stimulus

6.2 Experiments with a Real Robotic Agent

In Experiment III, the concepts of embodied intelligence were incorporated into a real robotic agent. The main idea of this experiment is to use the same control system developed for the simulated robotic agent in a real robotic agent. There was just a fine tuning of the input sensors

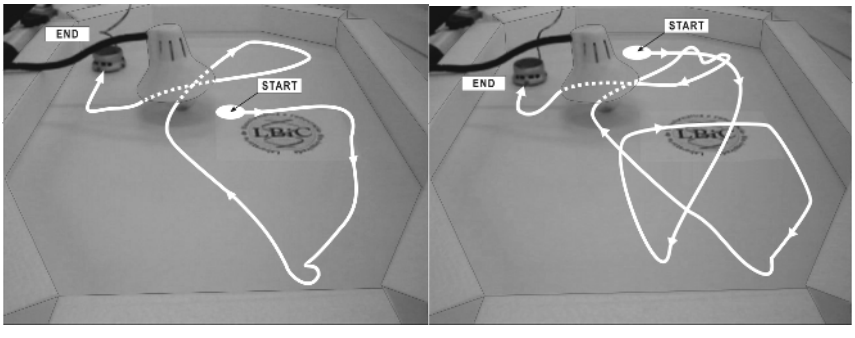


Fig. 6. Experiment III(a): A complete trajectory of the real Khepera robot, in an environment surrounded by walls, with a light source in the middle (b): An extended simulation from a different starting point.

Figure 6a shows a complete trajectory in an environment surrounded by walls, with a light source in the middle. The robot initiates the navigation searching for the nearest wall and starts to follow it. When its internal state level exceeds a predetermined limit, the artificial endocrine system determines the increase of the hormone level. This will cause the robot to follow the light (to recharge its battery). As soon as the robot reaches the light, the hormone level starts to decrease and the robot switches

back to the wall-follower behaviour. Figure 6b illustrates the same idea, but shows a higher number of behaviour changes from a different starting point.

7 Conclusion

This paper presented a biologically inspired artificial homeostatic module of an artificial endocrine system based on a previous work for robot autonomous navigation. It also contributed with fundamental principles and concepts for the novel approach towards the goal of creating artificially homeostatic (and self-organizing) systems.

The experiments conducted support the adaptability capacity of the artificial homeostatic system (simulation and experiments with real robots) through its ability to cope with internal and external changes, and also the ability to adapt to a dynamical environment by presenting a cyclic behaviour synchronized with the resources available. We made the robot react in synchrony with the environment without any internal world model and only using sub-symbolic representation. The results presented also show the basic operation of the control loop generally associated with homeostasis, where internal processes and external environmental factors are allowed to regulate the system.

The ultimate perspective is that the system proposed constitutes part of a new form of implementing embedded cognitive science. Comparative analysis of the current proposal and alternative Bio-inspired mechanisms to endow an artificial organism with autonomy is being considered as a further step of the research.

References

1. Besendovsky, H. O. & del Rey, A. (1996). Immune-Neuro-Endocrine Interactions: Facts and Hypotheses. Endocrine Reviews, 17(1), pp. 64-102.
2. Brooks, R. & Breazeal, C. (in press). Embodied intelligence. Cambridge, MA, MIT Press.
3. Brooks, R. (1991a). “Intelligence without representation”. AI, 47, pp. 139-160.
4. Brooks, R. (1991b). “Intelligence without reason”, in J. Mylopoulos and R. Reiter (Eds.) Proceedings of the 12th Int. Joint Conference on AI, Morgan Kaufmann.
5. de Castro, L. N. & Timmis, J. (2002). “The Immune System in Context with Other Biological Systems”, Chapter 5 in Artificial Immune Systems: A New Computational Intelligence Approach, Springer-Verlag: London.
6. Dorigo, M., Colombetti, M. (1997). Robot Shaping: An Experiment in Behavior Engineering (Intelligent Robotics and Autonomous Agents), MIT Press.
7. Floreano, D., Mondada F. (1995a). “Evolution of homing navigation in a real mobile robot”, IEEE Transactions on System, Man and Cybernetics.
8. Floreano, D., Mondada F. (1995b). “From evolution of innate behaviours to evolution of learning in robotics agents”, Technical Report R95.06I, Laboratory of Microcomputing, Swiss Federal Institute of Technology at Lausanne.
9. Greger, R. and Windhorst, U. (1996), (eds.) “Comprehensive Human Physiology” (Vols. 1 and 2). Springer.
10. Guyton, A. C. & Hall, J. E. (1996). Textbook of Medical Physiology, 9th ed., W. B. Saunders Company, Philadelphia.

11. Haykin, S. (1999). Neural Networks: A Comprehensive Foundation, 2nd ., Prentice Hall.
12. Klopf, A. H. (1982). The Hedonistic Neuron: a Theory of Memory, Learning, and Intelligence, Washington: Hemisphere.
13. K-Team S.A.. (2003). URL: http://www.k-team.com
14. McClintic,J. R. (1975). Basic Anatomy and Physiology of the Human Body,J.Wiley & Sons
15. McClintic, J. R. (1985). Physiology of the Human Body, 3rd ed., John Wiley & Sons
16. Neal M. and Timmis J. (2003). Timidity: A Useful Mechanism for Robot Control? Informatica, Vol. 27 No. 4, pp. 197-204.
17. Nolfi S. and Floreano, D. (2000). Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines, The MIT Press.
18. Perretta, S. J. and Gallagher, J.C. (2003). The Java Khepera Simulator from the Wright State University, Ohio, USA, http://ehrg.cs.wright.edu/ksim/ksim.html.
19. Pfeifer, R. and Scheier, C. (1999). Understanding Intelligence, MIT Press.
20. Purves W. K., Heller, H. C, Orians, G. H. and Sadava, D. (2001). Life: The Science of Biology, 6th Edition, IE-Macmillan UK.
21. Thagard, P. (1996). Mind: Introduction to Cognitive Science, The MIT Press, USA.
22. Timmis, J. and Neal, M. (2004). “Once More Unto the Breach: Towards Artificial Homeostasis”, in L. N. de Castro and F. .J. Von Zuben, Recent Developments in Biologically Inspired Computing, Idea Group Inc., Chapter XIV, pp. 340-366.

Artificial Life for Natural Language Processing*

Gemma Bel-Enguix and M. Dolores Jiménez-López

Research Group on Mathematical Linguistics,
Universitat Rovira i Virgili,
Pl. Imperial Tarraco, 1, 43005 Tarragona, Spain
gemma.bel@urv.net, mariadolores.jimenez@urv.net

Abstract. A framework for natural language processing based on an artificial life model is introduced. Human-computer interfaces require models of dialogue structure that capture the variability and unpredictability within dialogue. In this paper, taking as starting point the notion of eco-grammar system, and by extending it to the concept of Conversational Grammar Systems (CGS), we introduce a new formal framework for conversation modelling.

1 Introduction

Grammar systems theory is a consolidated and active branch in the field of formal languages [3]. Grammar systems have been defined as grammatical models of multi-agent architectures, stressing in this way the relationship between that theory and models coming from Artificial Intelligence. While grammar systems are related to AI, a subfield of the theory, –the so-called eco-grammar systems– is closely related to Artificial Life. Eco-grammar systems provide a syntactical framework for eco-systems, this is, for communities of evolving agents and their interrelated environment.

In this paper, we suggest the possibility of modeling dialogue by means of formal languages using eco-grammar systems. Eco-grammar systems present several advantages to account for dialogue: generation process is highly *modularised* by a distributed system of contributing agents; *contextualized*, linguistic agents re-define their capabilities according to context conditions given by mappings; and *emergent*, it emerges from current competence of the collection of active agents. Taking as starting point the notion of eco-grammar system, we introduce *Conversational Grammar Systems* (CGS)

Throughout the paper, we assume that the reader is familiar with the basics of formal language theory, for more information see [13] and [12].

2 Eco-Grammar Systems

Briefly, an eco-grammar system can be defined as a multi-agent system where different components, apart from interacting among themselves, interact with

* This research has been supported by a Marie Curie European Reintegration Grant (ERG) under contract number MERG-CT-2004-510644.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 765–774, 2005.

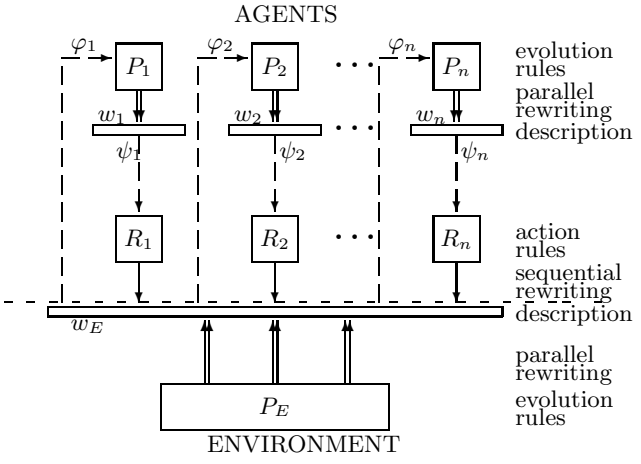


Fig. 1. Eco-Grammar System

a special component called 'environment'. So, within an eco-grammar system we can distinguish two types of components *environment* and *agents*. Both are represented at any moment by a string of symbols that identifies current state of the component. These strings change according to sets of evolution rules (L systems). Interaction among agents and environment is carried out through agents' actions performed on the environmental state by the application of some productions (rewriting rules) from the set of action rules of agents.

The concept of eco-grammar system is based on six postulates formulated according to properties of artificial life:

1. An ecosystem consists of an *environment* and a *set of agents*. Both state of the environment and states of agents are described by strings of symbols of given alphabets.
2. In an ecosystem there is a *universal clock* which marks time units, the same for all the agents and for the environment, according to which agents and environment evolution is considered.
3. Both environment and agents have characteristic *evolution rules* which are in fact L systems, hence are applied in a parallel manner to all the symbols describing agents and environment; such a (rewriting) step is done in each time unit.
4. Evolution rules of environment are *independent* on agents and on the state of the environment itself. Evolution rules of agents *depend on* the state of the environment (at a given moment, a subset of applicable rules is chosen from a general set associated to each agent).
5. Agents act on the environment according to *action rules*, which are pure rewriting rules used sequentially. In each time unit, each agent uses one action rule which is chosen from a set depending on current state of the agent.

6. *Action has priority over evolution* of the environment. At a given time unit exactly the symbols which are not affected by action (in the environment) are rewritten (in a parallel manner) by evolution rules.

Main features of eco-grammar systems are captured in Figure 1.

3 Conversational Grammar Systems: Basic Definitions

Eco-Grammar System has been defined as a grammatical model that tries to capture the distinctive features of an ecosystem, this is, a community of developing agents that interact with their dynamically changing shared environment. Within such a device we can distinguish two types of components: *environment* and *agents*. If we define an eco-grammar system as a syntactic model for a collection of cooperating agents that modify a shared common context and *dialogue* as maintained production of strings of mutually dependent acts built by two or more agents controlling and basing each other on the actions of the other ones. The analogy is quiter clear. However, eco-grammar systems lack some important features of conversation and this is the reason why we introduce Conversational Grammar Systems.

3.1 Context and Speakers

In any conversation we find two essential elements: context and speakers. Context must refer to factors relevant to understanding communicative behaviour and it is something shared by every participant in dialogue. It determines the actions speakers perform. But, at the same time, context is the addressee of such actions. We describe a dialogue as a *sequence of acts that change current context* (cf. [1]). Context –called *environment*– may be simply described as a string of symbols w_E, over an alphabet V_E. w_E contains any type of information necessary to develop the dialogue activity and it is shared by every agent in the system, this is, every participant in the talk exchange takes into account the state of w_E whenever it is to perform an action. w_E, at the same time, is changed during the dialogue act through speakers' actions. But, since not every aspect of context changes due to the speakers' utterances, we endow environment with a set of rules (P_E) that can be responsible for any change in the environmental string *not directly* produced by the agents' actions.

In order to have a dialogue, we need at least two participants, although more than two speakers may participate. On this basis, in CGS we have n *agents*, with $n \geq 2$. Each agent in the system is represented at any moment of time by a string of symbols w_i, over an alphabet V_i, $1 \leq i \leq n$. This string of symbols –representing the state of the agent– is modified during dialogue. Essentially, our *state* of knowledge at the end of the conversation is not the same as it was at the start. Therefore, we define a set of *rules* that accounts for the *evolution, modification or change* of the state of an agent. This finite set of rewriting rules over V_i^* is denoted by P_i, $1 \leq i \leq n$. However, since the state of speakers

in a conversation depends very much on the context where the interaction is taking place, we define a mapping which, on the basis of the current state of the environment, selects the rules from P_i that can be applied to an agent's state: $\varphi_i : V_E^* \rightarrow 2^{P_i}$. Moreover, participants need a mechanism that allows them to perform acts. Such a tool may be a set of rules R_i over V_E^*, called the set of *action rules* of i-th agent. Finally, we need to relate somehow the choice of actions to both the state of the context and the state of the agent. To do so, we can make use again of a mapping: $\psi_i : V_E^* \times V_i^+ \rightarrow 2^{R_i}$. The following definition captures what we have said up to now:

Definition 1. *A Conversational Grammar System (CGS) of degree n, $n \geq 2$, is an $(n+1)$-tuple:*

$$\Sigma = (E, A_1, ..., A_n),$$

where:

- $E = (V_E, P_E)$,
  - V_E *is an alphabet;*
  - P_E *is a finite set of rewriting rules over* V_E.
- $A_i = (V_i, P_i, R_i, \varphi_i, \psi_i, \pi_i, \rho_i)$, $1 \leq i \leq n$,
  - V_i *is an alphabet;*
  - P_i *is a finite set of rewriting rules over* V_i;
  - R_i *is a finite set of rewriting rules over* V_E;
  - φ_i*:* $V_E^* \rightarrow 2^{P_i}$;
  - ψ_i*:* $V_E^* \times V_i^+ \rightarrow 2^{R_i}$;
  - π_i *is the start condition;*
  - ρ_i *is the stop condition;*
  - π_i *and* ρ_i *are predicates on* V_E^*. *We can define the following special types of predicates. We say that predicate* σ *on* V_E^* *is of:*
    - *Type* (a) *iff* $\sigma(w) = true$ *for all* $w \in V_E^*$;
    - *Type* (rc) *iff there are two subsets* R *and* Q *of* V_E *and* $\sigma(w) = true$ *iff* w *contains all letters of* R *and* w *contains no letter of* Q;
    - *Type* (K) *iff there are two words* x *and* x' *over* V_E *and* $\sigma(w) = true$ *iff* x *is a subword of* w *and* x' *is not a subword of* w;
    - *Type* (K') *iff there are two finite subsets* R *and* Q *of* V_E^* *and* $\sigma(w) =$ *true iff all words of* R *are subwords of* w *and no word of* Q *is a subword of* w;
    - *Type* (C) *iff there is a regular set* R *over* V_E *and* $\sigma(w) = true$ *iff* $w \in R$.

The items of the above definition have been interpreted as follows: a) E represents the environment described at any moment of time by a string w_E, over alphabet V_E, called the *state of the environment*. The state of the environment is changed both by its own evolution rules P_E and by the actions of the agents of the system, A_i, $1 \leq i \leq n$. b) A_i, $1 \leq i \leq n$, represents an agent. It is identified

at any moment by a string of symbols w_i, over alphabet V_i, which represents its current state. This state can be changed by applying evolution rules from P_i, which are selected according to mapping φ_i and depend on the state of the environment. A_i can modify the state of the environment by applying some of its action rules from R_i, which are selected by mapping ψ_i and depend both on the state of the environment and on the state of the agent itself. Start/Stop conditions of A_i are determined by π_i and ρ_i, respectively. A_i starts/stops its actions if context matches π_i and ρ_i. Start/stop conditions of A_i can be of different types: (a) states that an agent can start/stop at any moment. (rc) means that it can start/stop only if some letters are present/absent in the current sentential form. And (K), (K') and (C) denote such cases where global context conditions have to be satisfied by the current sentential form.

Notice that definition 1 provides the necessary elements to maintain coherence in conversation. By limiting agents, in our system, to apply only rules included in ψ_i, we are forcing them to keep coherence. That is, by defining functions ψ_i and φ_i we are trying to ensure that actions performed by agents are connected to each other, thus preventing non-coherent sequences of actions.

3.2 Context-Change-Actions

CGSs intend to describe dialogue as a sequence of *context-change-actions* allowed by the current environment and performed by two or more *agents.* According to this idea, we define dialogue as a sequence of *acts* performed by two or more agents in a common environment. However CGS has to reflect the fact that actions in conversation are constrained by the *context*, by the *interlocutors' actions* and, of course, by our knowledge, beliefs, intentions, understanding, etc. In order to highlight all these facts, we have defined a mapping: $\psi_i : V_E^* \times V_i^+ \rightarrow 2^{R_i}$. With this mapping we have formalized the idea that not every action is allowable in dialogue. In the view of dialogue as a sequence of *context-change-actions* allowed by the current environment and performed by two or more agents, an *action* is defined as the application of a rule *on the environmental string*:

Definition 2. *By an action of an active agent A_i in state $\sigma = (w_E; w_1, w_2, \ldots, w_n)$ we mean a direct derivation step performed on the environmental state w_E by the current action rule set $\psi_i(w_E, w_i)$ of A_i.*

Definition 3. *A state of a CGS $\Sigma = (E, A_1, \ldots, A_n)$, $n \geq 2$, is an $n+1$-tuple:*

$$\sigma = (w_E; w_1, \ldots, w_n),$$

where $w_E \in V_E^*$ is the state of the environment, and $w_i \in V_i^*$, $1 \leq i \leq n$, is the state of agent A_i.

This rule is applied *by an active agent* and it is a rule selected by $\psi_i(w_E, w_i)$. We define an *active agent* in relation to the allowable actions it has at a given moment. That is, an agent can participate in conversation –being, thus, active– only if its set of allowable actions at that moment is nonempty:

Definition 4. *An agent A_i is said to be active in state $\sigma = (w_E; w_1, w_2, \ldots, w_n)$ if the set of its current action rules, that is, $\psi_i(w_E, w_i)$, is a nonempty set.*

Since conversation in CGS is understood in terms of *context changes*, we have to define how the environment passes from one state to another as a result of agents' actions:

Definition 5. *Let $\sigma = (w_E; w_1, \ldots, w_n)$ and $\sigma' = (w'_E; w'_1, \ldots, w'_n)$ be two states of a CGS $\Sigma = (E, A_1, \ldots, A_n)$. We say that σ' arises from σ by a simultaneous action of active agents $A_{i_1}, \ldots, A_{i_r}$, where $\{i_1, \ldots, i_r\} \subseteq \{1, \ldots, n\}$, $i_j \neq i_k$, for $j \neq k$, $1 \leq j, k \leq r$, onto the state of the environment w_E, denoted by $\sigma \overset{a}{\Longrightarrow}_\Sigma \sigma'$, iff:*

- *$w_E = x_1 x_2 \ldots x_r$ and $w'_E = y_1 y_2 \ldots y_r$, where x_j directly derives y_j by using current rule set $\psi_i(w_E, w_{i_j})$ of agent A_{i_j}, $1 \leq j \leq r$;*
- *there is a derivation:*
  $w_E = w_0 \overset{a}{\Longrightarrow}^*_{A_{i_1}} w_1 \overset{a}{\Longrightarrow}^*_{A_{i_2}} w_2 \overset{a}{\Longrightarrow}^*_{A_{i_3}} \ldots \overset{a}{\Longrightarrow}^*_{A_{i_r}} w_r = w'_E$

  such that, for $1 \leq j \leq r$, $\pi_{i_j}(w_{j-1}) = true$ and $\rho_{i_j}(w_j) = true$. And for $f \in \{t, \leq k, \geq k\}$ the derivation is:
  $w_E = w_0 \overset{a}{\Longrightarrow}^f_{A_{i_1}} w_1 \overset{a}{\Longrightarrow}^f_{A_{i_2}} w_2 \overset{a}{\Longrightarrow}^f_{A_{i_3}} \ldots \overset{a}{\Longrightarrow}^f_{A_{i_r}} w_r = w'_E$

  such that, for $1 \leq j \leq r$, $\pi_{i_j}(w_{j-1}) = true$[1], and
- *$w'_i = w_i$, $1 \leq i \leq n$.*

However, in the course of a dialogue, agents' states are also modified and the environmental string is subject to changes due to reasons different from agents' actions. So, in order to complete our formalization of dialogue development, we add the following definition:

Definition 6. *Let $\sigma = (w_E; w_1, \ldots, w_n)$ and $\sigma' = (w'_E; w'_1, \ldots, w'_n)$ be two states of a CGS $\Sigma = (E, A_1, \ldots, A_n)$. We say that σ' arises from σ by an evolution step, denoted by $\sigma \overset{e}{\Longrightarrow}_\Sigma \sigma'$, iff the following conditions hold:*

- *w'_E can be directly derived from w_E by applying rewriting rule set P_E;*
- *w'_i can be directly derived from w_i by applying rewriting rule set $\varphi_i(w_E)$, $1 \leq i \leq n$.*

In CGS, the development of dialogue implies that both the *state of the environment* and *state of agents* change. Such changes take place thanks to two different types of processes: *action steps* and *evolution steps*. By means of the former, active agents perform actions on the environmental string modifying its state; the latter imply the reaction of context and agents which, according to the changes produced by agents' actions, modify their states. So, action steps and evolution steps alternate in the course of dialogue. At the end, what we have is a *sequence of states* reachable from the initial state by performing, alternatively, action and evolution derivation steps:

[1] In this latter case the stop condition $\rho_i(w_j) = \text{true}$ is replaced by the stop condition given the f-mode.

Definition 7. *Let $\Sigma = (E, A_1, \ldots, A_n)$ be a CGS and let σ_0 be a state of Σ. By a state sequence (a derivation) starting from an initial state σ_0 of Σ we mean a sequence of states $\{\sigma_i\}_{i=0}^{\infty}$, where:*

- $\sigma_i \stackrel{a}{\Longrightarrow}_{\Sigma} \sigma_{i+1}$, *for* $i = 2j$, $j \geq 0$; *and*
- $\sigma_i \stackrel{e}{\Longrightarrow}_{\Sigma} \sigma_{i+1}$, *for* $i = 2j + 1$, $j \geq 0$.

Definition 8. *For a given CGS Σ and an initial state σ_0 of Σ, we denote the set of state sequences of Σ starting from σ_0 by $Seq(\Sigma, \sigma_0)$.*

The set of environmental state sequences is:

$$Seq_E(\Sigma, \sigma_0) = \{\{w_{Ei}\}_{i=1}^{\infty} \mid \{\sigma_i\}_{i=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_i = (w_{Ei}; w_{1i}, \ldots, w_{ni})\}.$$

The set of state sequences of the j-th agent is defined by:

$$Seq_j(\Sigma, \sigma_0) = \{\{w_{ji}\}_{i=1}^{\infty} \mid \{\sigma_i\}_{i=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_i = (w_{Ei}; w_{1i}, \ldots, w_{ji}, \ldots, w_{ni})\}.$$

$Seq(\Sigma, \sigma_0)$ describes the behaviour of the system, this is, the possible state sequences, directly following each other, starting from the initial state. $Seq_E(\Sigma, \sigma_0)$ and $Seq_j(\Sigma, \sigma_0)$ are the corresponding sets of sequences of the states of the environment and of the states of j-th agent, respectively.

Now, we associate certain languages with an initial configuration:

Definition 9. *For a given CGS Σ and an initial state σ_0 of Σ, the language of the environment is:*

$$L_E(\Sigma, \sigma_0) = \{w_E \in V_E^* \mid \{\sigma_i\}_{i=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_i = (w_E; w_1, \ldots, w_n)\}.$$

and the language of j-th agent is:

$$L_j(\Sigma, \sigma_0) = \{w_j \in V_A^* \mid \{\sigma_i\}_{i=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_i = (w_E; w_1, \ldots, w_j, \ldots, w_n)\}.$$

for $j = 1, 2, \ldots, n$.

$L_E(\Sigma, \sigma_0)$ and $L_j(\Sigma, \sigma_0)$ correspond to those states of the environment and to those states of the j-th agent, respectively, that are reachable from the initial configuration of the system.

3.3 Selection Techniques

Two important selection techniques in dialogue are the turn-taking system and the adjacency pairs. If we want to provide a formal language account of turn-taking, we should focus on the most important traits of this phenomenon, and make it susceptible to formalization. It is possible to reduce turn-taking to the *stop* of a participant and the *start* of another one. But, since this stopping/starting is carried out with no gap and no overlap, it should be necessary to endow agents with some mechanism that allows them to recognize when they can start/stop their contributions without overlapping and with no gap between them. If we want to guarantee turn-taking we should, firstly, make sure that whenever an agent starts to perform actions, it will stop. In order to do so, we define different *derivation modes* that control how long an agent can act in the environmental state:

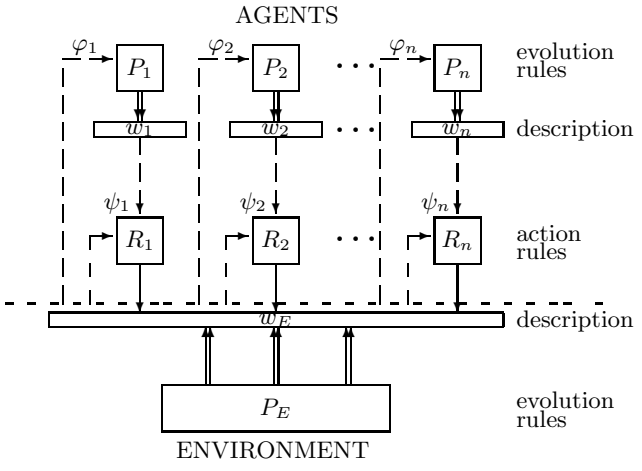


Fig. 2. Conversational Grammar Systems

Definition 10. *Let $\Sigma = (E, A_1, ..., A_n)$ be a CGS. And let $w_E = x_1x_2...x_r$ and $w'_E = y_1y_2...y_r$ be two states of the environment. Let us consider that w'_E directly derives from w_E by action of active agent A_i, $1 \leq i \leq n$, as shown in Definition 5. We write that:*

$w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{\leq k} w'_E$ *iff* $w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{\leq k} w'_E$, *for some* $k' \leq k$;
$w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{\geq k} w'_E$ *iff* $w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{\leq k} w'_E$, *for some* $k' \geq k$;
$w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{*} w'_E$ *iff* $w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{k} w'_E$, *for some* k;
$w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{t} w'_E$ *iff* $w_E \overset{a}{\underset{A_i}{\Longrightarrow}}^{*} w'_E$ *and there is no* $z \neq y$ *with* $y \overset{a}{\underset{A_i}{\Longrightarrow}}^{*} z$.

In words, $\leq k$-derivation mode represents a time limitation where A_i can perform at most k successive actions on the environmental string. $\geq k$-derivation mode refers to the situation in which A_i has to perform at least k actions whenever it participates in the derivation process. With $*$-mode, we refer to such situations in which agent A_i performs as many actions as it wants to. And finally, t-derivation mode represents such cases in which A_i has to act on the environmental string as long as it can.

One way of getting transitions with no gap and no overlap in CGS is to endow agents with an *internal control* that contains start/stop conditions that allow agents to recognize places where they can start their activity, as well as places where they should stop their actions and give others the chance to act. This is, start/stop conditions help agents to recognize *transition relevance places*, i.e. places where speaker change occurs. Start/stop conditions have been formally defined in Definition 1.

It seems quite common in talk exchanges to find paired actions. Notions such as adjacency pairs, reactive pressures, discourse expectations etc. intend to account for the fact that utterances produced in dialogue are somehow determined and constrained by preceding utterances in the talk exchange. Mapping

$\psi_i(w_E, w_i)$ fulfils in CGS a function analogous to the one carried out by all the above notions in their respective conversational models. This mapping establishes which actions are allowed for agent A_i at any given moment. Notice that $\psi_i : V_E^* \times V_i^+ \rightarrow 2^{R_i}$ does not contain a *single* rule allowed for A_i at a given moment, but, on the contrary, it specifies a *set* of permitted actions from which A_i can choose one to apply on the environmental string.

3.4 Closings

Closing a dialogue implies that participants stop their conversational activity *because they have reached their goal* in the talk exchange. For deciding when the computation terminates, we have to determine which string is to be considered as the reference point to signal the end of the derivation. It seems that it should be possible to identify various cases in which the derivation process in a CGS is finished. We can consider, for example, that derivation is finished if there is *some* agent that has reached a terminal string or just in the case when *every* agent in the system has a terminal state. And we can take into account, as well, such cases in which derivation is ended if there is an *identified* agent that has already got a terminal string. Summing up, we can identify at least three different styles of closing derivation process in CGS:

Definition 11. *Let $\Sigma = (E, A_1, ..., A_n)$ be a CGS as in Definition 1. Derivation in Σ terminates in:*

- *Style (ex) iff for $A_1, ..., A_n$, $\exists A_i : w_i \in T_i$, $1 \leq i \leq n$;*
- *Style (all) iff for $A_1, ..., A_n$, $\forall A_i : w_i \in T_i$, $1 \leq i \leq n$;*
- *Style (one) iff for $A_1, ..., A_n$, $A_i : w_i \in T_i$, $1 \leq i \leq n$.*

According to the above definition, a derivation process ends in style *(ex)* if there is *some* agent A_i that has reached a terminal string. It ends in style *(all)* if *every* agent in the system has a terminal string as state. And it finishes in style *(one)* if there is *one* distinguished agent whose state contains a terminal string. Styles *(all)*, *(ex)* and *(one)* might account for three different ways of closing a dialogue. They may refer, respectively, to such cases in which a dialogue is closed by mutual agreement of all the participants, those in which some speakers decide to end the exchange irrespectively of the fact that the rest of speakers still have something to say, and those in which one identified participant is holding conversation and can close the talk interaction as soon as he/she has fulfilled his/her goals.

4 Final Remarks

In this paper we have introduced some elements for a formal modelling of human communication using eco-grammar systems. Basic issues related to an eco-grammar system approach to conversation have been developed in order to test

the suitability of the model. Nevertheless, since this is just an initial approximation to the possibility of describing conversation by means of an artificial life model, many important aspects remain to be approached. Anyway, we claim that conversational grammar systems are able to model dialogue with a high degree of flexibility, what means that they are able to accept new concepts and modify rules, protocols and settings during the computation. Evolution and action are involved in a consistent way in environment/contexts, while interaction of agents with the medium is constant. Moreover, conversational grammar systems present the following advantages to account for conversation:

- generation process is highly *modularised* by a distributed system of contributing agents;
- it is *contextualized*, linguistic agents re-define their capabilities according to context conditions given by mappings;
- and *emergent*, it emerges from current competence of the collection of active agents.

References

1. Bunt H.C.: Belief Context in Human-Computer Dialogue, in Nauta, D., Nijholt, A. & Schaake, J. (eds.): *Pragmatics in Language Technology*, TWLT4, University of Twente (1993) 106-114.
2. Crow, B.K.: Topic Shifts in Couples' Conversations, in Craig, R. & Tracy, K. (eds.): *Conversational Coherence. Form, Structure, and Strategy*, Sage Publications, California (1983) 136-156.
3. Csuhaj-Varjú, E., Dassow, J., Kelemen, J. & Păun, Gh.: *Grammar Systems: A Grammatical Approach to Distribution and Cooperation*, Gordon and Breach, London (1994).
4. Csuhaj-Varjú, E., Kelemen, J., Kelemenová, A. & Păun, Gh.: Eco-Grammar Systems: A Grammatical Framework for Life-Like Interactions, *Artificial Life*, 3, 1 (1996) 1-28.
5. Gibbs, Raymond & Mueller, Rachel A.G.: Conversation as Coordinated, Cooperative Interaction, in Zachary, W., Robertson, S.P. & Black, J.B. (eds.): *Cognition, Computation, and Cooperation*, Ablex, New Jersey (1990) 95-114.
6. Jiménez-López, M.D.: *Using Grammar Systems*, GRLMC Report, Rovira i Virgili University, Tarragona (2002).
7. Jiménez-López, M.D.: Dialogue Modeling with Formal Languages, in Spoto, F., Scollo, G. & Nijholt, A. (eds.): *Algebraic Methods in Language Processing*, TWLT 21, University of Twente (2003) 207-221.
8. Kelemen, J. & Kelemenová, A.: A Grammar-Theoretic Treatment of Multiagent Systems, *Cybernetics and Systems*, 23 (1992) 621-633.
9. Lee, J.R.E.: Talking Organization, in Button, G. & Lee, J.R.E. (eds.): *Talk and Social Organization*, Multilingual Matters LTD, Clevedon (1987) 19-53.
10. Leech, G. N.: *Principles of Pragmatics*, Longman, London (1983).
11. Levinson, S.C.: *Pragmatics*, Cambridge University Press, Cambridge (1983).
12. Rozenberg, G. & Salomaa, A.: *Handbook of Formal Languages*, Springer, Berlin (1997).
13. Salomaa, A.: *Formal Languages*, Academic Press, New York (1973).
14. Wardhaugh, R.: *How Conversation Works*, Blackwell, Oxford (1985).

A Co-evolutionary Epidemiological Model for Artificial Life and Death

Alan Dorin

Centre for Electronic Media Art, Computer Science & Software Engineering,
Monash University, Clayton, Australia 3800
aland@csse.monash.edu.au

Abstract. This paper presents a model of the co-evolution of transmissible disease and a population of non-randomly mixed susceptible agents. The presence of the disease elements is shown to prevent the onset of genetic convergence of the agent population. The epidemiological model also acts in a distributed fashion to counter the tendency of the agent population to occupy spatially close-knit communities. The simulation applies a modified mathematical SIR epidemiological model of disease transmission in combination with the well-studied technique of artificial ecosystems. It includes various aspects of disease transmission that are not usually modelled due to the effort required to incorporate them into mathematical models. These include a distributed agent population with non-uniform infectiousness and immunity as well as a mutable disease model with evolving latency and infections that evolve to prey on diverse agent characteristics.

Keywords: Epidemiological model, co-evolution, artificial death, ecosystem.

1 Motivation and Past Work

Digital evolutionary simulations may converge with the population predominantly of similar genetic composition. This convergence is undesirable if it occurs (for example, due to the presence of local maxima in the fitness landscape) before a solution to an optimisation problem has been discovered. Hence where the desired simulation result is an exploration of diverse solutions within some constraints, early convergence needs to be avoided. This is also the case when the problem is to generate an interesting, evolving agent population of sonic or visual forms for an artwork or interactive installation [1]. For this latter reason it was decided to explore a strategy for preventing the convergence of a population that would be applicable generally to agent simulations.

To meet its aim, this paper prefers an elegant, emergent, decentralized approach over that of a hard-coded or centralized controller. Hence, it focuses primarily on a notable omission from many Artificial Life models and publications, *disease*. Typical Artificial Life ecological simulations model creatures competing for food, mating, fighting, and dieing. Yaeger's *PolyWorld* is a seminal example in which agents interact utilizing colour vision [2]. Todd has noted strategies for removing creatures from

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 775 – 784, 2005.

a population subject to a genetic algorithm but stops short of exploring different reasons for their death [3] (for example disease or suicide). Mascaro et al. have dealt specifically with suicide in a population of simple agents [4]. Ray's *Tierra* simulation eliminates elderly or ineffective population members with a "reaper". Also of interest is the emergence of "parasitic" code in his system [5]. The usual methods of removing members of a population may also have been employed – culling random agents, unfit agents or replacing parents with their offspring for example. These standard approaches do not improve the diversity of the population at any one time, in fact if carelessly applied they may be responsible for its convergence.

The Artificial Life literature has much to say on co-evolution as a means of improving a genetic algorithm's performance through increased population diversity [6, 7]. This work is similarly inspired, only the simulation models virtual worlds and does not optimise explicit fitness functions. The co-evolutionary model presented here is novel also since the disease/parasite that co-evolves with the agent population is wholly dependent for its position in space (and in fact its existence) on the susceptible agents.

1.1 An Introduction to Models from Epidemiology

A fully-cited history of the mathematical theory of epidemics is beyond the scope of this paper. The history leading to the classic model discussed below is provided in [8, 9].

At least since the 1920's, stochastic models of epidemics have been utilized. The standard model is based on a population of individuals who are either susceptible to a specific disease (*susceptibles* denoted **S**) or infected with the disease and capable of transmitting it to others (*infectives* denoted **I**). Population members who overcome a disease may become immune to further infection[1] or may become susceptible once again depending on the particular disease. Population members who are immune to a disease or remain infected but through isolation cannot transmit it, are considered *removed* (denoted **R**). The model as described is known as an *SIR* model. It may be modified slightly to provide fresh susceptibles through birth or immigration.

Some pertinent parameters of epidemic models are as follows. The period of time during which a disease exists entirely within an organism is known as the disease's *latent period*. The organism is not infective during this period. An *incubation period* often follows latency. During incubation the organism may not show outward sign of infection but is nevertheless infective. Usually once the incubation period is over, the victim of the disease is clearly marked by symptoms and can therefore be avoided by susceptibles.

Probabilistic epidemiological models that operate in discrete time steps are particularly suited to implementation in software.[2] At any time step, the probability of a new case of the disease appearing is proportional to the number of susceptibles multiplied

[1] Following a bout of a disease a victim may be deceased, alternatively their immune system may prevent repeat infiltration by the same virus.

[2] It is interesting to note that in the 1920's two American epidemiologists Reed and Frost demonstrated a discrete *mechanical* model in which coloured balls represented susceptibles and infectives.

by the number of infectives. This basic model assumes random mixing of individuals in the population and does not allow for the complex interactions between physically separated sub-populations, nor for variable incubation or latent periods of a disease. The problems inherent in models that make simplifying assumptions concerning the nature of spatial distributions are discussed in [10]. Various extensions to the SIR model to allow for these phenomena have been added over the last fifty years. Some mathematical models and computer simulations deal with the spatial distribution of susceptibles along a line, across a lattice or over a network to overcome the inaccuracies due to the assumption of random mixing of the population. Cellular-automata and other discretized versions of the SIR method have been utilized also [11, 12]. Some of these models have also incorporated disease *carriers* (e.g. some viruses are transferred by mosquito), and non-homogeneous populations. The model presented in the current paper allows all of these phenomena to emerge from the simulation without hard-coding their behaviour.

The current threats of biological warfare and terrorism have raised the stakes in Western society for epidemiology. The U.S. National Institute of General Medical Sciences has devoted $1.6 billion to a fledgling agent-based study of epidemics [13]. Like the U.S. project, this paper adopts agent modelling to represent the principles of epidemiology in an intuitive but realistic fashion. As shall be shown, the process of epidemic spread offers a means of increasing the genetic and phenotypic diversity of a population and of capping its density.

1.3 Relevant Consequences of Basic Epidemic Theory

There are two theories of epidemiology that are particularly relevant here. The first of these is known as the *Threshold Theorem* [14]: a disease cannot take hold in a population of susceptibles unless the population density is above a particular threshold. This value relates to the infectivity of a disease and the death and recovery rates it induces. If population density passes beyond the threshold, the disease will reduce the population to a level as far below the threshold as it was above it prior to the epidemic.

The *Threshold Theorem* has many consequences, one of which has come to be known as *Herd Immunity* [8, pp. 27-31]. This theory indicates that a calculable number less than the full population needs to be immunized to prevent an epidemic. Unfortunately the theory has been shown to provide inaccurate figures in practice, due to its assumption of random mixing in a population. Nevertheless, it highlights an important aspect of epidemics, namely that the spread of a disease is not dependent on the percentage of a population who are immune, but on the contact between susceptibles and infectives. When a population does not mix uniformly, the supply of susceptibles may be similarly irregular.[3] The model presented in this paper does not assume random mixing of a population, rather the agent interactions are emergent from the simulation.

[3] For example, if a socio-economic group is immunized against a disease, and these people do not mix randomly with people from other groups, an epidemic may still occur within the latter groups whilst the former is immunized. I.e. sub-group mixing is *important* in considering the spread of a disease.

2 An Agent-Based Simulation of Infectious Disease Epidemics

The present simulation runs in discrete time steps during which a population of agents moves freely about a continuous-space, virtual world. The model was originally devised as a part of a generative, interactive artwork (described elsewhere [15]) that exhibits numerous emergent features typical of Artificial Life simulations. The model's essential features are described below.

2.1 Agent Composition, Behaviour and Evolution

Agents are represented visually as coloured boxes. These have a position and velocity on a continuous toroidal surface. Each agent may wander randomly over the space at a speed inversely proportional to its volume. During each time step of the simulation, agents expend an amount of energy proportional to their volume to move and metabolise. At each simulation time step, energy is gained by an agent from the environment in an amount proportional to its upper surface area as if each box-top was equipped with a solar cell charging a battery. Agents exhausting their energy supply "die" and are removed from the simulation. Agents also age throughout a simulation and are removed if they reach the end of their lifespan.

Agents perceive their neighbours' positions, dimensions and colours within a limited visual range. An agent may accelerate towards (or away from) a neighbour that it finds attractive (or repulsive) as determined by reference to colour and dimension templates it stores. Each agent stores colour and dimension templates marking properties it finds attractive in its partners and templates marking repulsive properties. The closer the match of a particular template the greater the tendency of the agent to seek or flee the neighbour that exhibits it. These tendencies are used to adjust the velocity of the agent as it moves.

If two agents' bodies intersect one another, find one another attractive, and pass a maturity/age threshold test, they may produce a single offspring agent per time step at their current location. The offspring is initiated with energy donated by each of the parents. This donation costs parents an amount of energy specified in their property list. The characteristics of the offspring are specified by the crossover and mutation of the parents' genotypes. This is an array of floating-point values coding the properties listed in Table 1. The system employs a single crossover point and mutation of one gene in every offspring by a random amount between +/- 5%.

New births are subject to an overflow test of the available simulation space. If a birth would cause an overflow the request is refused. Following an unsuccessful request, a random member of the population may be eliminated from the simulation to make room for future requests.

Table 1. Floating-point agent genotype contents (italics indicate vector quantities)

Colour (R,G,B)	*Colour preference*	*Colour abhorrence*
Dimension (X,Y,Z)	*Dimension preference*	*Dimension abhor-*
Visual range	Offspring energy dona-	Lifespan

2.2 Disease Behaviour and Evolution

The agents in the model may carry virtual diseases, transmit them to other agents and succumb to infection themselves. The diseases in the simulation co-evolve alongside the agent population but may only exist *within* a host agent i.e. disease does not persist in the environment. A susceptible agent is exposed to a disease when it intersects with an infective agent. An agent that is carrying a disease cannot be infected by a second disease (i.e. an active disease blocks secondary infection).

If an agent is not carrying a disease, its susceptibility is determined by the match between its own colour and the *colour-signature* of the carried disease to which it is brought into contact. The closer the match between the agent's colour and the colour-signature template of the disease, the higher the probability the disease will infect the susceptible agent during a time step of contact. Simulation diseases also possess a *devastation* value that measures the virulence of a disease. This parameter is used to scale the probability of infection and the amount of energy required of a host to survive a time step of infection.

A parameter determines the lifespan of a simulation disease in each host. Long-lived diseases require a host to invest substantial amounts of energy to overcome infection. If a disease is overcome without the death of the host, the agent acquires immunity to the strain of the disease by adding it to an *immunity list*. Any further contact with this disease will result in an *immune response* that prevents the disease from infecting the agent a second time. If a disease kills its host, or the host dies for any reason, the disease it carries dies also, irrespective of its lifespan.

Each disease has parameters determining its *latent* and *incubation* periods (see section 1.1). A latent disease does not require energy of its host and is not infectious. During the incubation period the agent is infective but does not exhibit symptoms. Agents may visually detect disease symptoms in neighbours, potentially allowing them to steer clear, however this feature was not utilized in the current experiments.

Real diseases such as viruses replicate and mutate within a host much more rapidly than the hosts themselves reproduce, circumventing the host's auto-immune response. Consequently, it is possible for humans to repeatedly catch viruses such as the common cold and flu. To model this, a simulation disease undergoes reproduction during every time step of its lifespan. Disease reproduction is asexual and may result in mutation of the disease parameters: colour-signature; devastation; lifespan; incubation and latent periods. A parameter that sets the frequency of a disease's mutation during reproduction may itself be mutated. Together these parameters allow the diseases to co-evolve with the more slowly evolving agent population.

Diseases are represented in the simulation as coloured shapes rendered within the box bodies of the agents. Fig. 1 illustrates the visualization scheme employed.

The parameters for the disease and agents outlined fully specify the features of a epidemic models discussed above. A complex and flexible simulation has been devised that allows for studies of epidemics in non-homogeneous populations with non-random mixing. This agent-based model eliminates many of the problems inherent in earlier epidemiological models.

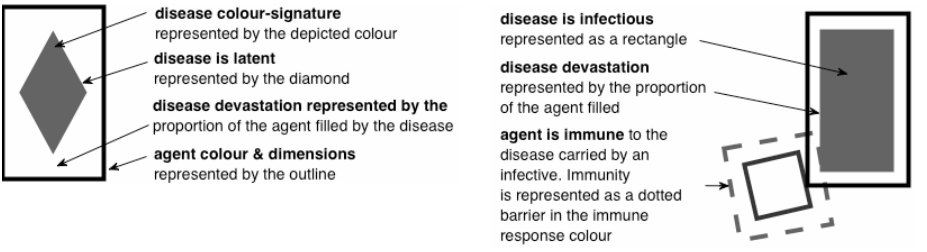


Fig. 1. The visualization scheme for agents and infection

3 Results

As indicated in the motivation for this work, it had been noted that ecological simulations in which agents competed for resources (including mates and energy) often resulted in a genetically impoverished, homogeneous population. It was hoped that by introducing a novel co-evolutionary model of disease, diversity might be encouraged and uniformity exploited and eliminated by infection.

3.1 Qualitative Discussion

It was found that the disease did indeed exploit the population's uniformity when it arose. Disease also exploited populations of agents that clustered tightly together. In the absence of disease, agents of particular colours and sizes often dominated a simulation, forming large colonies of potential mates. A typical screen shot after 14,000 time steps of the simulation without disease is reproduced in fig. 2(a). Fig. 2(b) illustrates a run after 14000 time steps with identical initial conditions, but in which the disease model was introduced. The diversity in dimensions, colour and spread of the population is far greater in fig. 2(b) than in fig. 2(a). In fact, after as few as 2500 time steps, the non-diseased model often converges to homogeneity and does not break from this condition but drifts gently through genetic space. The population model incorporating disease maintains its diversity indefinitely.

A disease simulation run involves the spontaneous appearance of a disease on average once every one-hundred-thousand agent updates. This new disease is generated with a colour-signature that matches the colour of a randomly selected agent. The agent is infected with the disease and left to continue its travels. Apart from the colour-signature, all other disease parameters for the new infection are randomly generated.

Depending on the parameters of the new disease, the traits of the infected agent and the population as a whole, the new disease may or may not cause an epidemic. The likelihood of an epidemic is specified by the Threshold and Herd Immunity theories described above. Some observed outcomes are described below along with the conditions giving rise to them in the present simulation environment.

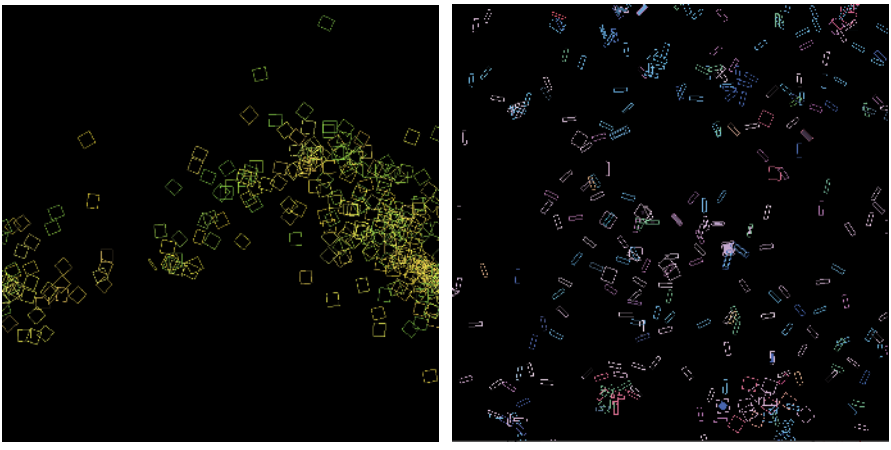


Fig. 2. Two simulation screenshots after 14,000 time steps: (a) without the epidemiological model; (b) with the epidemiological model.

Disease elimination (immediate). If the disease is insufficiently long-lived, or the population is insufficiently dense, or the host does not co-habit with others of a similar colour to itself, then the disease may fail to contact any susceptibles before it dies within the host. The disease will be eliminated from the population immediately.

Disease spread (immediate). A disease may mutate sufficiently within a host to infect susceptibles of a colour significantly different to the original host. If the host mixes amongst others of its kind they may become infected with the disease also. Occasionally the stochastic mechanism allows for a disease to infect a host coloured differently to its own signature. In this case, the devastation of the disease will be low in the infected host but the host nevertheless is able to infect other susceptibles. Such a host may be considered a "carrier" of the disease.

Disease elimination (eventual). If the disease manages to take a hold in the population it may nevertheless die out eventually if the number of susceptibles is reduced. This may happen when a sizeable proportion of the agents encountered by infectives is immune to the disease (even though the population as a whole may not have a significant number of immune members – see footnote 3 above). Circumstances like this arise when agents overcome the disease and acquire immunity, or when the disease is so devastating that it rapidly wipes out the supply of susceptibles before the agents are able to produce many offspring.

Disease spread (continual). A disease well-suited to its environment has sufficient lifespan to ensure it is passed from one susceptible agent to another. Such a disease also needs to be sufficiently devastating that it can be transferred successfully, but not so devastating that it kills off its supply of susceptibles. Diseases that fit these criteria also have to be sufficiently stable to avoid unwanted mutations that would render them ineffective, but sufficiently mutable so that they can keep infecting an evolving population of hosts. The simulation has given rise to diseases that meet all of these criteria and persist in the population for long periods of time.

Of particular interest are diseases that sustain themselves indefinitely when they are able to utilize susceptibles that are prolific breeders. Such diseases are able to spread through contact between mates who seek one another out (sexually transmitted diseases?) and also by contact between a parent and its newly born. Newly born agents may have traits slightly different to their parents so that occasionally one tends to wander off to seek its own preferred companions, taking the disease to infect others. As long as the disease remains latent for a sufficiently long interval, it will not kill or weaken the agent prior to its immigration to a further enclave.

3.2 Gene Diversity Plot Analysis

Figure 3 gives example plots of the red colour gene value of each agent in the population, versus the simulation time step for (a) a healthy agent population and (b) a population in which disease is present. Each simulation commences with a population of randomly generated agents, and therefore figures 3(a) and 3(b) show red gene values to be widely spread at time step 0.

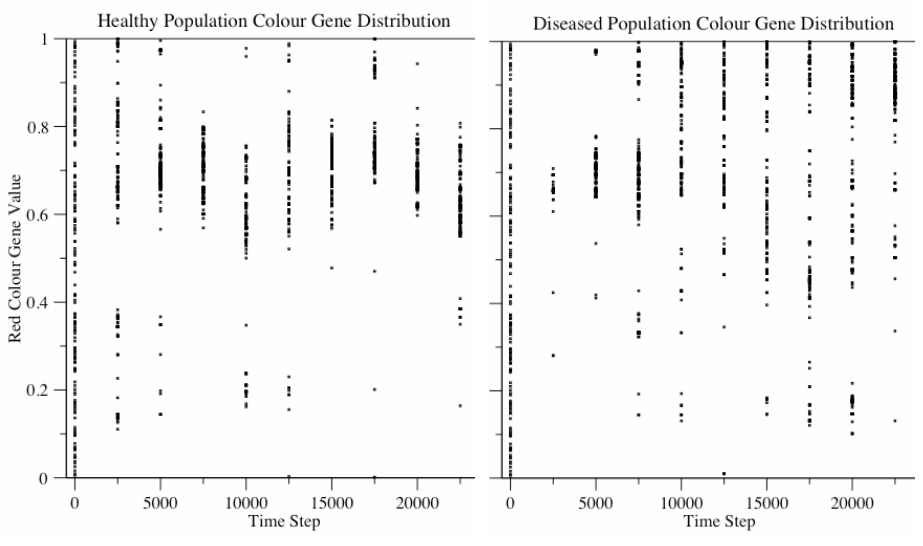


Fig. 3. Plots of all agents' red colour gene values against simulation time step: (a) without the epidemiological model; (b) with the epidemiological model

Time step 2500 of plot 3(a) commences a long-term decline in the diversity of the red gene in the population. Without the presence of disease, the combination of colour genes an agent possesses determines its mating success based on the presence of potential mates who find the colour of the agent attractive. Thus, agent colour in the disease-free population is driven purely by its ability to attract mates. The decline in diversity is visible as the vertical dispersal of the red colour gene in the population is reduced over time. The few "outliers" at each time step are excursions into new colours brought out by a momentary success of a sub-population with a specific colouration. Such events may be the result of spontaneous mutations during reproduction. The main population drift in figure 3(a) has red gene values focussed from 0.5 to 0.8.

Figure 3(b) shows the red gene diversity in the diseased population. After the initial random spread of red gene values, the diversity in the population declines dramatically by time step 2500 to a range limited between 0.6 and 0.7. This was the result of a few randomly introduced diseases culling a population that had not yet adapted to existence *without* the presence of disease, i.e. they could not yet locate mates and produce offspring efficiently. A few rapidly acting random diseases therefore wiped out much of the population before it had a chance to evolve strategies for sustaining itself. The system has been programmed to generate several offspring automatically from two randomly selected parents (even if they are not close to one another) in situations like this in order to "jump start" the simulation. This has the drawback of starting a population with a limited gene pool.

The situation depicted in figure 3(b) is especially interesting because with the presence of disease, even this limited gene pool (at time step 2500) does not simply drift about genetic space as did the disease-free simulation when it encountered homogeneity. Instead, as can be seen from subsequent time steps of figure 3(b), the diversity of the population actually expands. The co-evolutionary pressure between the disease and the agents ensures that this situation is maintained indefinitely.

Figure 3 can therefore be seen to confirm the discussion at the beginning of this section and the interpretation of figure 2 given above. In summary, the disease acts to maintain colour diversity in the population, despite pressure applied by mating preferences to the contrary. The disease also forces the population to spread across the available space and it allows the agents to explore a wider variety of shapes than the pressure of the environment alone would have permitted. Both these latter results are clearly depicted in figure 2.

4 Conclusions and Future Work

A model of epidemics has been introduced to an evolutionary, agent-based simulation. The model improved the overall diversity of the population as desired and also encouraged its spread across the available virtual space. A wide variety of disease outcomes emerged from the simulation, each an apparently plausible model of real-world outbreaks.

Future work of interest to the author is a full investigation of the impact of the Threshold Theorem utilizing the present simulation. Can its behaviour be predicted mathematically and demonstrated successfully using this model? It would also be interesting to conduct experiments that model known infectious diseases and their dispersal based upon known interactions of animal or human populations.

References

1. Dorin, A.: The Virtual Ecosystem as Generative Electronic Art. 2nd European Workshop on Evolutionary Music and Art, Applications of Evolutionary Computing: EvoWorkshops 2004. Coimbra, Portugal, Springer-Verlag (2004) 467-476
2. Yaeger, L.: Computational Genetics, Physiology, Metabolism, Neural Systems, Learning, Vision and Behavior or Polyworld: Life in a New Context. Artificial Life III, Addison-Wesley (1992) 263-298

3. Todd, P.M.: Artificial Death. Jahresing 41 (German modern art yearbook) (1994) 90-107
4. Mascaro, S., Korb, K.B., Nicholson, A.E.: Suicide as an Evolutionary Stable Strategy. Advances In Artificial Life, 6th European Conference. Prague, Springer (2001) 120-133
5. Ray, T.S.: An Approach to the Synthesis of Life. Artificial Life II. Santa Fe, New Mexico, Addison Wesley (1990) 371-408
6. Cartlidge, J., Bullock, S.: Caring Versus Sharing: How to Maintain Engagement and Diversity in Coevolving Populations. 7th European Conference on Artificial Life. Dortmund, Germany, Springer-Verlag (2003) 299–308
7. Hillis, W.D.: Co-Evolving Parasites Improve Simulated Evolution as an Optimization Procedure. Artificial Life II. Santa Fe, New Mexico, Addison-Wesley (1990) 313-324
8. Ackerman, E., Elvebeck, L.R., Fox, J.P.: Simulation of Infectious Disease Epidemics. Charles C Thomas Springfield, Illinois (1984)
9. Bailey, N.T.J.: The Mathematical Theory of Infectious Diseases and Its Applications. 2. Charles Griffin & Company London (1975)
10. Durrett, R., Levin, S.: The Importance of Being Discrete (and Spatial). Theoretical Population Biology 46:3 (1994) 363-394
11. Willox, R., Grammaticos, B., Carstea, A.S., Ramani, A.: Epidemic Dynamics : Discrete-Time and Cellular Automaton Models. Physica A 328 (2003) 13-22
12. Martins, M.L., Ceotto, G., Alves, S.G., Bufon, C.C.B., Silva, J.M., Laranjeira, F.F.: A Cellular Automata Model for Citrus Variagated Chlorosis. eprint arXiv : cond-mat/0008203 (2000) 10
13. Pilot Projects for Models of Infectious Disease Agent Study (Midas). Webpage. National Institute of General Medical Sciences (NIGMS) http://grants.nih.gov/grants/guide/rfa-files/RFA-GM-03-008.html (Accessed: October 2004)
14. Kermack, W.O., McKendrick, A.G.: Contributions to the Mathematical Theory of Epidemics. Proceedings of the Royal Society, London 115 (1927) 700-721
15. Dorin, A.: Artificial Life, Death and Epidemics in Evolutionary, Generative Electronic Art. In Rothlauf, F. (ed.) Evoworkshops 2005, Springer-Verlag, Berlin; Heidleberg (2005) 448-457

CoEvolution of Effective Observers and Observed Multi-agents System

Christophe Philemotte[1] and Hugues Bersini[2]

[1] IRIDIA - Université Livre de Bruxelles, 1050 Brussels, Belgium
cphilemo@iridia.ulb.ac.be
[2] IRIDIA - Université Livre de Bruxelles, 1050 Brussels, Belgium
bersini@ulb.ac.be

Abstract. This paper elaborates upon an idea and a development introduced and presented by Bersini in [1]. Roughly, by observing the search space of a combinatorial problem in a "clever" way, it can be drastically reduced. In order to discover this "clever way", a second search process has to be engaged in the space of the observables. So two Genetic Algorithms (GAs) are intertwined to solve the whole problem: one in the original space and one in the space of observables of the original one. We are going to present and evaluate this idea on a Cellular Automata (CA) implementation of a binary numbers adder. The experiments show that the new algorithm, combining the two evolutionary searches, speeds up the research and/or increases the quality of the solutions in a significant way.

1 Introduction

The essential part of research about evolutionary algorithms or any optimisation algorithm in general is the discovery of mechanisms to improve the search, when the problem is characterized by a huge search space. Many metaheuristics and hybridisations of those algorithms are invented and compared with their capacity to traverse this search space in the most effective way. One alternative approach when facing the problem of the space dimension is to discover some clever ways to reduce it. The notion of "intrinsic emergence" originally inspired by the developments of Crutchfield and Mitchell appears to be very helpful [2].

According to them and other authors [2,3,4,5,6], and as further discussed in [1], a macro property, labelled as "emergent", should supply some mechanical and non-human observer with additional functional meaning. This "functional device" replaces the common need of a human observation to characterize emergence [7,8,9,10,11,12]. Indeed, as shown in [1], this concept offers an interesting way to encode macroscopically the genome of a multi-agent system, and by doing so, to reduce temporally the size of the search space. In [1], the different ways to observe the search space were tested randomly, while here, for greater coherence, the search in the "space of observables" extends the idea and uses an evolutionary mechanism. This combination of the two evolutionary searches is the core of the new algorithm presented in this paper.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 785–794, 2005.

To test the effectiveness of this approach, the problem treated here is the evolution and the discovery of a CA able to simulate a binary adder. This problem is chosen for different reasons. First of all, CA is the favorite computational platform to illustrate emergent phenomena [8,10,13,14]. Secondly, the generalisation of the new approach to other multi-agents systems might be easy: masking the influence of agents neighbourhood to update an agent. A final major reason is the engineering interest in binary addition. Indeed, this task is more relevant than "density classification" or "synchronisation" [3,4,5,15] for computer applications.

Section 2 will describe in detail the experimental platform (the CA) and the task to be performed. In Section 3, we give more details on the simple evolutionary algorithm applied to discover efficient CA. These principles are both used for the classical GA and for the extension we propose: the Genetic and Emergent Algorithm (GEA). Then, in Section 4, we define the macro observable to emerge and show how the complete GEA works. Finally, in Sections 5 and 6, we present the main results and show how the added evolution of the observable subsequently accelerates the discovery of a satisfactory solution.

2 CA and Binary Addition

The task of the CA, which has been chosen to illustrate the algorithm, is to perform the binary addition. This choice is mainly motivated by its engineering relevance which might lead to new engineering practices in the construction of logical circuits.

The CA used here is bi-dimensional with periodic boundaries and characterized by the classical 8-cells Moore neighbourhood. The state domain is the binary set $\{0, 1\}$ and the state update law is synchronous. This update law composed of the CA rules table is coded in a binary array. It is composed by all possible 2^8 update cellular cells which are indexed by the corresponding neighbourhood. We adopt a non-uniform version of CA which offers more computational power than the uniform case [15]. Each cell is then characterized by a "variety" v and updated by the corresponding rules table [13].

Considering n the bit string length, the CA used is a square of $n^2 \times n^2$ cells. We set at time 0, in the first line, the two n bits binary terms: the first one on the left and the second one on the right. All other cells are initialized to `0`. Then, we compute n^2 time steps of the CA by obeying the rules table explained above. To perform the addition task, a $n + 1$ bits encoding of the sum of the two previous numbers have been obtained in the bottom left corner. Finally, the whole problem consists in achieving this addition task for m given couples of binary numbers. The process of addition defined above is very reminiscent of classical addition logical circuits: we set the numbers at one side, they cascade through a sequence of logical gates, and we finally take out the result on the other side. We have no guarantee of obtaining a universal adder but just a specific one for m additions. However and very naturally, the larger m, the more universal the adder.

3 Adopted Evolutionary Strategy

Note that we are only interested in the whole idea of the intrinsic emergence and thus the precise evolutionary strategy adopted here to find the good solution is not relevant. This same idea can be applied to any search and evolutionary algorithm for multi-agent or combinatorial problems.

The individual, namely the non-uniform CA, is genetically represented by its evolution law, the different varieties of rules. Hence, the length of the chromosome, which is composed of the v bit strings representing the different rules, is $v2^8$: a huge search space of dimension 2 to the power $v2^8$. To fit every candidate, we sum Hamming distance between the exact solution and the sum obtained by evolved the CA and this over each addition problem. Thus, the fitness belongs to the integer interval $[0, m(n+1)]$. We decide to use the same evolutionary method as the one used in [1], the "elitist version". After evaluating every CA performing m binary additions, the individuals are ranked by decreasing fitness. We select the half best population $s_r/2$. The best individual is preserved without any genetic change in the next generation. The second and the third offspring are generated by randomly mutating two individuals selected randomly in the population. One bit of their chromosomes is randomly chosen and flipped. The next $(s_r - 3)$ offspring are obtained by applying the simplex crossover on three individuals randomly selected from the population. This crossover method, first introduced in [16], has been shown to improve on the classical one. This strategy is adopted here for the classical GA.

4 A Useful Way to Observe CA

As already discussed, even if the variety parameters is limited to three values, the space cardinality is still around 10^{231}, making it hard for a classical GA to find the global optimum in a decent time. Based on the intrinsic emergence concept, an original way to observe the CA is discussed in [1]: the macro observable consists in masking some of the eight neighbours of all cells to be updated. As a result the search space is considerably reduced. Consequently, at any update step of the CA, the algorithm does not take into consideration the masked neighbours in order to compute the new state. For instance, in Figure 1 a) a specific mask of five cells is seen. Suppose the neighbourhood state to be `01100101`, the masked version is compressed to `101`.

The CA working is not modified by excessive simplification. Indeed, the mask determines a set of neighbourhood states that gives an identical result following the given rules (and its variety). Only the unmasked cells state is important. This fact allows us to compress the rules coding. In our previous example, for instance, the following neighbourhood states give the same result: `01100101`, `01110100`, `01000111`, `11010110`, `11100101`, ... any `b1bb01bb` where the masked bits b are not taken into consideration. Thus, we can strongly reduce the space needed for coding the rules and therefore the whole search space. The mask acts on the search space by prohibiting some areas as shown in Figure 1 b), hence

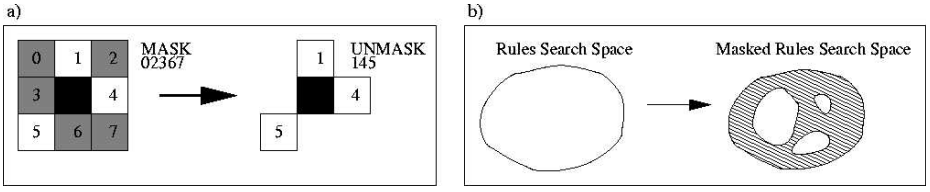


Fig. 1. a) The black cell is the cell to update. The grey ones are the masked cell neighbours and the white ones are the unmasked cell neighbours. We have here a mask imposed on five cells reducing the neighbourhood to a 3-cells one. b) Interpretation of the action of the mask in the rules search space.

restricting the search to the permitted regions. If the number of masked cells is denoted by l_m, we have $(8 - l_m)$ unmasked cells and the rules are coded on $v2^{8-l_m}$. The size of the search falls down to 2 to the power $v2^{8-lm}$. For instance, by considering 3 varieties and 5 masked cells, the space cardinality collapses to around 10^7.

Now all masks are certainly not similarly adequate to observe the CA since some masks may turn out to hide the region where the global optimum really takes place. Thus, it is important to find the best mask among the $C_8^{l_m}$ possibilities. This approach does not defer the combinatorial explosion on the mask search. As shown in [1], this new way of observation trims the problem before actually searching for a more precise solution. We explain in the next section how to automate the search of a good mask and how to have the best "intrinsically emerged" one.

Before presenting the GEA, the main topic of the paper, some explanations might be useful about how to transform the reduced rules table, i.e. the rules used in the CA masked version, into the complete rules table and conversely. The reduced complete rule table transition is achieved in three steps. First, we generate all the possible 2^8 bit strings. Second, we define a function to project the unmasked neighbour binary number on the masked version. Finally, for each variety, we convert the update value of the unmasked rules table to the masked one.

5 How to Evolve the Mask: The Genetic Emergence Algorithm (GEA)

The mask constitutes a macro observable of the system, abstracting or skipping unnecessary details during some periods of the optimisation process. The algorithm below presents the complete process. The mask is submitted to the evolutionary operations following the feedback of the system: it is selected by the system itself, and the best one will intrinsically emerge with no need for human intervention. In comparison with [1], the improvement proposed in this paper consists in obtaining the best mask, no anymore by random trials, but by the same evolutionary process as the one searching for the optimal rule table.

We have here another instance of a co-evolution process where the observable evolves and emerges in the same time as the CA.

To evolve the mask, a given population of corresponding masking rules is evolved and the fitness of the mask is evaluated by computing the fitness of the rules set obtained under this masking condition. After given iterations both the mask and the population of the best rules obtained with this mask are memorized in order to start a new set of simulations in which the complete coding (i.e. we take back the 8 neighbours) of the cell states and the rules are re-established. The following pseudo code describes the working of our GEA into three sequential steps:

- **The initialisation phase:**

```
RULES_POP: RANDOM INIT
CA: RANDOM INIT OF VARIETY OF CA CELLS
MASK_POP: RANDOM INIT
```

- **The masking phase:** a mask population is evolved. To perform this task, each mask has to be evaluated by evolving the corresponding masked rules population (see Figure 2), defining following parameters:
  - g_m the evolution time for the mask,
  - s_m the size of the mask population,
  - g_{mr} the evolution time for masked rules,
  - s_r the size of the rules population (masked and unmasked), and
  - g_r the evolution time for unmasked rules.

```
FOR tm = 0 TO g_m
   FOR i = 0 TO s_m
      MASK_RULES_POP: RULES_POP+MASK_POP[i]
      FOR tmr - 0 TO g_mr
         FOR j = 0 to s_r
            EVALUATE MASK_RULES_POP[j]
         ENDFOR
         SELECT BEST HALF MASK_RULES_POP
         GENERATE NEW MASK_RULES_POP
      ENDFOR
      EVALUATE MASK_POP[i]
      SAVE TMPBESTMASK_RULES_POP
   ENDFOR
   SELECT BEST HALF MASK_POP
   SAVE BESTMASK_RULES_POP
   GENERATE NEW MASK_POP
ENDFOR

RULES_POP: SAVE BESTMASK_RULES_POP
```

- **The classical phase:** using the best masked rules set, a new unmasked rules population is built. This population is then evolved by a classical GA.

```
FOR tr = 0 TO g_r
   FOR k = 0 to s_r
      FOR m PROBLEM
         EVALUATE RULES_POP[k] ON CA
      ENDFOR
   ENDFOR
   SELECT BEST HALF MASK_RULES_POP
   GENERATE NEW MASK_RULES_POP
ENDFOR
```

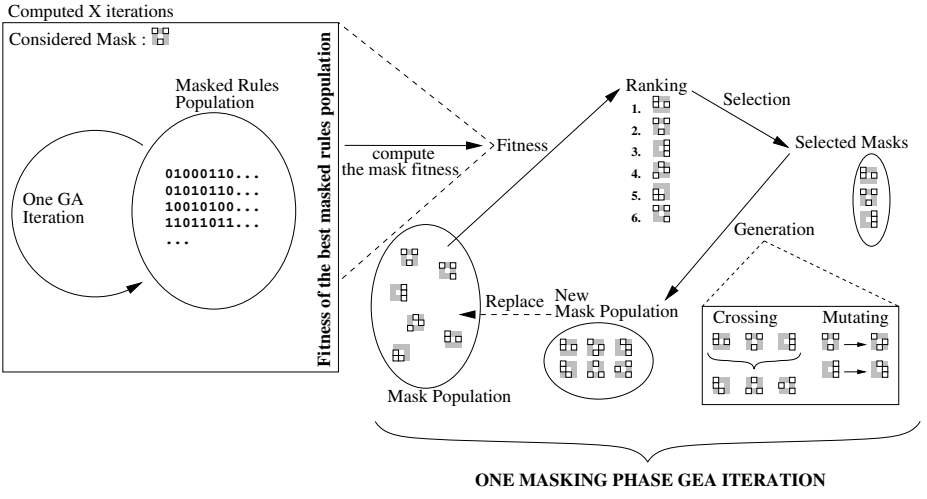


Fig. 2. The mechanism of the GEA masking phase

So, we can approximately evaluate the number of GA iterations needed in order to have an equivalent running complexity as for the GEA:

$$I_{GA} = g_r + s_m g_m g_{mr}$$

Before presenting the results, we must clarify how the fitness of a mask is computed on the basis of the fitness of the corresponding masking rules population. The best way, obtained after various trials, is to compute the product between the mean of the fitness of all rules and the best fitness (i.e. the lowest). Other methods have been tried (only the mean or the best), but the following simple method gives a faithful estimation of the quality of the population: both on average and with respect to the best individual.

6 GA vs GEA

In order to verify in a fair way the average benefit in time and in fitness offered by GEA, the results need to be compared to a GA running without any mask.

Four kinds of experiment are conducted for each algorithm, for five different varieties (1 to 5). Here, the common parameters are:

- a CA of 25×25 cells ($n = 5$) and an CA evolution on 25 steps,
- five non-uniform rules sets ($v \in \{1, 2, 3, 4, 5\}$),
- ten addition problems chosen randomly ($m = 10$), where the addition terms is coded on 5 bits and the solution on 6 (big endian coding),
- a population of rules of 20 individuals ($s_r = 20$),
- a population of masks of 6 individuals ($s_m = 6$),
- and a mask length of 5 ($l_m = 5$, we thus have 56 possible masks),

Four different sets of the "time" parameters (i.e. g_m, g_{mr} and g_r) are chosen:

1. **Experiment 1, Figure 3 a):** $g_m = 5$, $g_{mr} = 2$, $g_r = 10$ and $I_{GA} = 70$, runs 100 times;
2. **Experiment 2, Figure 3 b):** $g_m = 10$, $g_{mr} = 7$, $g_r = 70$ and $I_{GA} = 490$, runs 50 times;
3. **Experiment 3, Figure 4 a):** $g_m = 25$, $g_{mr} = 8$, $g_r = 200$ and $I_{GA} = 1400$, runs 50 times;
4. **Experiment 4, Figure 4 b):** $g_m = 5$, $g_{mr} = 14$, $g_r = 70$ and $I_{GA} = 490$, runs 50 times.

Obviously, the size of the mask search space is ridiculously small here for the discovery of the optimal mask. We are however only in its generalisation ability for any size of the masks and any size of the original search space. This is the reason we chose to evolve the mask. The motivations behind each choice are given below:

1. **Experiment 1:** How do GEA and GA react for a few number of iterations ?
2. **Experiment 2:** Being sure to test nearly all mask possibilities, what is the benefit of GEA ?
3. **Experiment 3:** Does the GEA confirm its improvement for a greater number of iterations even if the search in mask space is too long for the cardinality of the mask space ?
4. **Experiment 4:** Proper use of the evolutionary strategy on mask, is the GEA more powerful than the GA ?

The results are shown in Figure 3 and Figure 4. For each experiment, the mean fitness is given over all simulations obtained for the GA, the masking phase of GEA (noted GEA1) and the complete GEA (noted GEA2). We remember that the lowest the fitness is, the best the CA is. The three differences between these results are: GA-GEA1, GA-GEA2 and GEA1-GEA2. With these figures, each main question find answers and some remarks can be added.

According to the first experiment, the fitness obtained with GEA is 25% better on average than with GA except for the uniform CA, in terms of quality. Due to the small number of iterations, there is no real gain between the GEA masking phase and the GEA classical phase. This is because the GEA classical phase does not have enough time to converge. Furthermore, there are also no significant differences between the different non-uniform case.

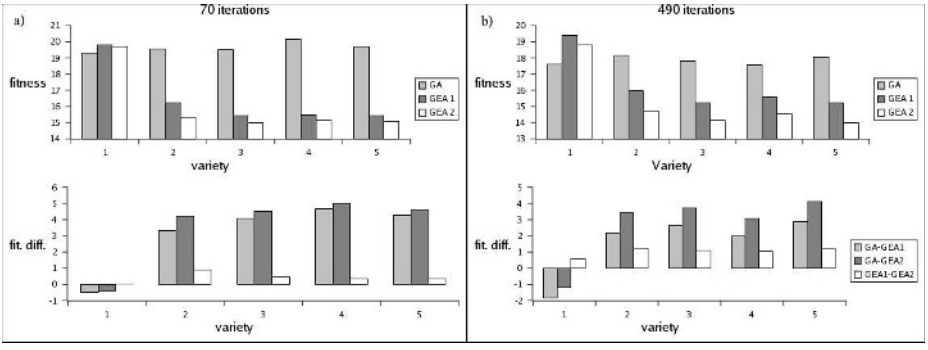


Fig. 3. a) Results for experiment 1. Mean fitness (over 100 simulations) is plotting against CA varieties. The first graph shows the results obtained after the GA, the GEA masking phase (GEA1) and the complete GEA (GEA2). The second graph gives the differences between these 3 functions: GA-GEA1, GA-GEA2 and GEA1-GEA2. b) Results for experiment 2. Mean fitness over (50 simulations) is plotting against CA varieties. The first graph shows the results obtained after the GA, the GEA masking phase (GEA1) and the complete GEA (GEA2). The second graph gives the differences between these 3 functions: GA-GEA1, GA-GEA2 and GEA1-GEA2.

Concerning the second experiment, we are sure to test nearly all possible mask. We are sure to find the optimal mask, i.e. the best macro observation. The improvement is confirmed: about 20% in quality of solution is obtained except for the uniform case. The GEA gives again worse results than GA for the uniform case. Probably because it is too much masked for the size of the rules

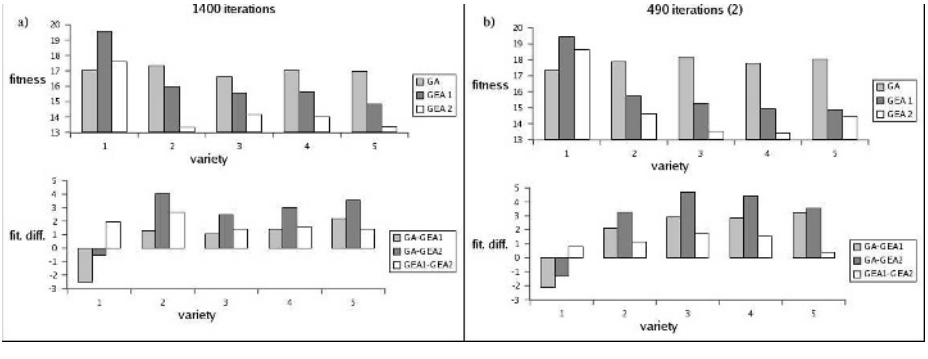


Fig. 4. a) Results for experiment 3. Mean fitness (over 50 simulations) is plotting against CA varieties. The first graph shows the results obtained after the GA, the GEA masking phase (GEA1) and the complete GEA (GEA2). The second graph gives the differences between these 3 functions: GA-GEA1, GA-GEA2 and GEA1-GEA2. b) Results for experiment 4. Mean fitness (over 50 simulations) is plotting against CA varieties. The first graph shows the results obtained after the GA, the GEA masking phase (GEA1) and the complete GEA (GEA2). The second graph gives the differences between these 3 functions: GA-GEA1, GA-GEA2 and GEA1-GEA2.

search space. Because more time is spent in the GEA classical phase, a better solution is obtained than after the masking phase. This fact is very interesting, because the number of iterations used by the classical phase represents only a seventh of the whole for the complete GEA. The convergence is thus faster: the optimal mask gives a good area where the solution can be searched.

The results of third experiment confirm the improvement: about 20% of improvement except again for the uniform case. We notice that the difference GA-GEA1 is this time worse than in the two previous experiments. It is certainly because we spend too much time in the masking phase to find the optimal mask. The masks are too much evolved for the size of its search space and the masking rules population is not sufficiently evolved to evaluate accurately the observation quality. The best convergence of the GEA classical phase is again confirmed after finding the optimal mask. Finally, similar results are obtained for the classical GA than for GEA in the first experiment: an impressive improvement in time.

The fourth experiment yields similar results to the second experiment. the "intrinsic emergence" concept is well represented by the chosen parameters because the mask is really evolved this time. The coevolution of the system macro-observable really improves the classical GA.

7 Conclusions

In this work, we have examined in more details the concept of "intrinsic emergence" from an engineering point of view. To do so, how to find a good CA implementation of a binary adder is the problem that has been chosen. To solve this problem, a classical GA can be used. However, this choice does not give good results due to the large size of the search space. As it was shown in [1], "intrinsic emergence" can be helpful to improve the classical GA.

A macro observable, a mask, represents then the intrinsic emergent property. By evolving the mask, the machine observer is fulfilled: this is the foundation of the main subject of this paper, the GEA. Indeed, added to a GA, we obtain a coevolution mechanism which is an improvement of the classical evolution strategy. To show this improvement, the efficiency of a classical GA is experimentally compared with the GEA on a given problem.

The results obtained through 4 experiment sets confirm the improvement of this new approach (see Figure 3 and Figure 4): a gain in time or in fitness for all non-uniform cases. So, making intrinsic system emergent property is a good way to reduce the search space and to boost the adaptative capacity of the CA population.

This work is a first practical interest of the "intrinsic emergence", and a lot of future work remains to be done. The concept of macro-observables could be extended to other kind of multi-agent systems and more generally to all optimisation problems which can be represented by bit string chromosome: masking becomes grouping given loci where the genes take the same allele. This will permit a true generalisation of the GEA and a starting point to a theoretical formalisation of the algorithm. Other ideas such as incremental masking or het-

erogeneous masking could be also interesting to test. But all these works must test this concept through a larger space of problems. By doing so, we will be able to better understand the subjacent basis of the discussed concept and the effect of the parameters on the efficiency of GEA.

References

1. Bersini, H.: Whatever emerges should be intrinsically useful. In: Artificial life 9, The MIT Press (2004) 226–231
2. Crutchfield, J.: Is anything ever new? considering emergence. In G. Cowan, D.P., Melzner, D., eds.: Integrative Themes. Volume XIX of Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley Publishing Company, Reading, Massachusetts (1994)
3. Packard, N.: Adaptation toward the edge of chaos. Dynamic Patterns in Complex Systems (1988) 293–301
4. Andre, D., Bennett III, F.H., Koza, J.R.: Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. In Koza, J.R., Goldberg, D.E., Fogel, D.B., Riolo, R.L., eds.: Genetic Programming 1996: Proceedings of the First Annual Conference, Stanford University, CA, USA, MIT Press (1996) 3–11
5. Crutchfield, J., Mitchell, M.: The evolution of emergent computation. In: Proceedings of the National Academy of Science. Volume 23. (1995) 103
6. Cariani, P.: Emergence of new signal-primitives in neural networks. Intellectica **2** (1997) 95–143
7. Baas, N.: Emergence, hierarchies, and hyperstructures. In Langton, C., ed.: Artificial life III. Volume XVII of Santa Fe Institute Studies in the Sciences of Complexity., Reading, Massachusetts, Addison-Wesley Publishing Company (1994) 515–537
8. Bedeau, M.A.: Weak emergence. In Tomberlin, J.E., ed.: Philosophical Perspectives: Mind, Causation, and World. Volume 11. Blackwell Publishers, Oxford (1997) 375–399
9. Emmeche, C., Koppe, S., Stjernfelt, F.: Explaining emergence: Towards an ontology of levels. Journal for General Philosophy of Science **28** (1997) 83–119
10. Kubík, A.: Toward a formalization of emergence. Artificial Life **9** (2003) 41–65
11. Poundstone, W.: The Recursive Universe. Chicago: Contemporary Books (1985)
12. Rasmussen, S., Baas, N.A., Mayer, B., Nilsson, M., Olesen, M.W.: Ansatz for dynamical hierarchiesr. Artificial Life **7** (2001) 329–353
13. Wolfram, S.: Cellular Automata and Complexity. Addison-Wesley Publishing Company, Reading, Massachusetts (1994)
14. Ronald, E.M.A., Sipper, M., Capcarrère, M.S.: Design, observation, surprise! a test of emergence. Artificial Life **5** (1999) 225–239
15. Sipper, M.: Computing with cellular automata: Three cases for nonuniformity. Physical Review E **57** (1998) 3589–3592
16. Bersini, H., Seront, G.: In search of a good evolution-optimization crossover. In Männer, R., Manderick, B., eds.: Second Conference Parallel Problem Solving from Nature, Amsterdam, NorthHolland, Elsevier Science Publishers (1992) 479–488

Comparative Reproduction Schemes for Evolving Gathering Collectives

A.E. Eiben, G.S. Nitschke, and M.C. Schut

Computational Intelligence Group,
Department of Computer Science, Faculty of Sciences,
De Boelelaan 1081a, 1081 HV Amsterdam,
The Netherlands
{gusz, nitschke, schut}@cs.vu.nl

Abstract. This research investigates an evolutionary approach to engineering agent collectives that accomplish tasks cooperatively. In general, reproduction and selection form the two cornerstones of evolution and in this paper we study various reproduction schemes in an evolving population of agents. We classify reproduction schemes in temporal and spatial terms, that is, by distinguishing when and where agents reproduce. In terms of the temporal dimension, we tested schemes where agents reproduce only at the end of their lifetime or multiple times during their lifetime. In terms of the spatial dimension we distinguished locally restricted reproduction (agents reproduce only with agents in adjacent positions) and panmictic reproduction (when an agent can reproduce with any other in the environment). This classification leads to four different reproduction schemes, which we compare, via their overall impact upon collective performance. Results using two completely different types of evolvable controllers (hand-coded or neural-net based) indicate that utilizing single reproduction at the end of an agent's lifetime and locally restricted reproduction afforded the agent collective a significantly higher level of performance in its cooperative task.

1 Introduction

The research theme of this paper is described by the term: *Emergent Collective Intelligence* (ECI). The end goal of ECI research is to combine and exceed achievements in multi-agent systems [1], swarm intelligence [2], and evolutionary computation [14] research via developing synthetic methodologies such that groups of computationally complex agents produce desired emergent collective behaviors resulting from the bottom-up development of certain individual properties and social interactions. This paper investigates certain technical aspects of artificial evolution as means of achieving adaptability at the local level and desired emergent behavior at the global level.

In many multi-agent tasks, such as those common to multi-robot systems, the correct input-output mappings for the agents' controllers are not known in advance so it is not possible to program or train them with supervised learning methods [3]. To solve this problem many researchers have used neuro-evolution [4] as a generalized

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 795 – 804, 2005.

methodology for adaptability in agent behavior. Many researchers have highlighted that neuro-evolution is most appropriately applied to complex problems that are neither effectively addressed via pure artificial evolution nor neural processing approaches [5], [6], [7]. For example, Gomez [8] devised the enforced sub-populations (ESP) neuro-evolution methodology that was used for deriving the correct input-output mappings for the agents' controllers in the learning of multi-agent control tasks [9] with small numbers of agents. ESP has been shown to work well for various discrete-state applications such as the game *Go*, as well as the classical pole-balancing task [10]. Neuro-evolution approaches investigated in collective robotic systems such as RoboCup soccer [11], simulated pursuit-evasion tasks [12], and multi-agent computer games [13] were only able to derive limited forms of cooperative behavior, and the behavior did not scale with the number of agents.

Our application domain is the gathering of renewable resources from an environment. This gathering task is divided into *locating*, *retrieving*, and *transporting* the resources in question. It is an essential assumption that this task is interfaced to the population of agents via fitness rewards that are given after delivering the resources to a given 'home area'. Additionally, we distinguish resources with different values and postulate that gathering of higher value (more complex) resources necessitates a higher degree of cooperative behavior (more agents). The performance evaluation criterion for the agent collective as a whole is then the *total value gathered cooperatively*, measured at the final generation of the simulation. Clearly there are many particular applications fitting into this general description. One could think of collecting some renewable resources, for example: harvesting farming produce, or minesweeping. We use the minesweeping example throughout this paper. That is, we consider the task of the agent collective to be the location, and extraction of different types of mines, and their transportation to the home area within a simulated mine field. The successful delivery of a mine to the home area is equated with a fitness reward and fitness rewards are proportional to the type and amount of mines gathered.

Our approach to developing successful agents for this task is evolutionary; in particular, we evolved agent controller parameter values. Hence, we studied two different types of agent controllers, one heuristic controller with evolvable parameters, and a neural net controller with the same set of evolvable parameters and evolvable connection weights. The technical research goal of this paper was to compare the efficacy of different agent reproduction scheme settings for accomplishing the minesweeping task. We classified reproduction schemes in temporal and spatial terms, that is, by distinguishing when, with which agents a given agent reproduces. For the temporal dimension, the agent reproduction schemes we tested were termed: *Single Reproduction at the End of the Agent's Lifetime* (SREL) and *Multiple Reproductions During an Agent's Lifetime* (MRDL). For the spatial dimension, we distinguished locally restricted reproduction (agents reproduce only with agents in adjacent positions) and panmictic reproduction (when agents reproduce with other agents located anywhere in the environment). This classification led to four different reproduction schemes, which we compared experimentally, using the collective performance of the population accomplishing its task as the basic measure.

2 Collective Behavior Design

The experiments utilized a simulated minefield and an initial population of 1000 agents, placed at random positions on a grid-cell environment with a 50 x 50 resolution. A maximum of four agents could occupy any given grid-cell within the environment. A home area spanning 4 x 4 grid-cells was randomly placed within the environment. This home area was where gathered mines were taken. Gathering was the term applied to the process of locating, extracting, transporting, and delivering a mine to the home area. Within the simulated minefield there were three types of mines: *type A*, *type B* and *type C*. The different types of mines had differing *values* to reflect the difficulty (degree of cooperation) associated with gathering it. The cost of gathering mines comprised two sub-costs: the cost of extracting a mine from its location in the environment, and the cost of transporting a mine to the home area. The costs of extracting and transporting one unit of each of the three mine types are presented in *table 1*. The transport cost was applied per unit being transported, and per grid-cell traversed. Initially, a quantity of between 0 and 3 mines of each type were randomly initialized and placed within each grid-cell. It is assumed that a long-term process of gathering and replenishment in a minefield is being simulated, where mines are considered a renewable resource, and each mine type is renewed at a rate of 3 per simulation iteration. That is, the simulation is of a long-term process of collective gathering behavior being evolved, whilst an unseen competitor renews gathered mines. Additionally, it is assumed that an agent never triggered a mine to detonate.

In order to gather the different mine types a degree of cooperative behavior was necessitated. Cooperation was necessary when at least one agent was attempting to extract a given mine type, and the value of the prevalent agent controller parameter was too low for the agent to individually gather the mine. These prevalent agent controller parameters were termed: *Mine type A capacity*, *Mine type B capacity*, *Mine type C capacity* and *transport capacity*, and provided an indication of the capability of an agent for gathering a particular mine type. Specifically, to gather *one unit* of a particular mine type, the sum of the values of the *capacity* parameter for that mine type (for all agents simultaneously attempting to extract the mine) must exceed a given *capacity threshold*. These capacity thresholds are presented for each mine type in *table 1*. The task of each agent was to gather the highest possible value of mines during the course of its lifetime. This task was interfaced to the agent collective by providing fitness rewards for gathered mines.

The fitness rewards for gathering one unit of the different mine types are presented in *table 1*. The total value of mines that all agents gathered in cooperation with at least one other agent during the course of its lifetime was termed the *value gathered cooperatively*. Further to playing its conventional role in survivor selection, fitness was also used as a metaphor of energy (actions cost fitness). An agent was able to move one grid-cell in any direction per simulation iteration at a cost of one unit of fitness.

Table 1. The capacity thresholds, and the costs for extracting and transporting mines, as well as the fitness reward for gathering one unit of the different mine types

	Capacity Threshold	Extraction Cost	Transport Cost	Fitness Reward
Mine type A	**300**	**8**	**0.04**	**20**
Mine type B	**150**	**4**	**0.02**	**10**
Mine type C	**75**	**2**	**0.01**	**5**

Value Ranges	Initial	Minimum to Maximum
Parameters: Not Evolvable		
Sight	1	1
Death_Age	[20..100]	[20..100]
Min_Fit_Reproduction	50	50
Parameters: Evolvable		
Mine Type A Capacity (CA)	[0..100]	[0..Infinity]
Mine Type B Capacity (CB)	[0..100]	[0..Infinity]
Mine Type C Capacity (CC)	[0..100]	[0..Infinity]
Transport Capacity (CT)	[0..300]	[0..Infinity]

Fig. 1.

```
IF AmA < CA THEN
  IF ( Holding + AmA ) < CT THEN extract AmA
ELSE IF AmB < CB THEN
        IF ( Holding + AmB ) < CT THEN extract AmB
  ELSE IF AmC < CC THEN
          IF ( Holding + AmC ) < CT THEN extract AmC
    ELSE Look-Ahead
```

```
Look-Ahead:
IF end of life and SREL active THEN reproduce
IF at home THEN unload mines transported
  IF MRDL active THEN reproduce
IF transporting a quantity of mines THEN move to home
ELSE IF mine type A detected THEN move to mine type A
  ELSE IF mine type B detected THEN move to mine type B
    ELSE IF mine type C detected THEN move to mine type C
      ELSE move to a random cell
```

Fig. 2.

Fig. 1. Evolvable and non-evolvable agent controller parameters. **Fig. 2.** Heuristics utilized by agents operating under the pure-evolution approach. *AmA*, *AmB*, and *AmC* denote the amount of mine type *A*, *B* and *C*, respectively, on a given grid-cell. *Holding* denotes the current amount of all mine types a given agent is currently transporting. *CA*, *CB*, *CC* and *CT*, denote the gathering capacities for mine types *A*, *B*, and *C*, and the transport capacity, respectively.

2.1 Pure-Evolution Approach

For the evolution of agent controller parameter values, a standard evolutionary algorithm was used [14]. When an agent initiated reproduction, the fittest partner (with the highest energy) of *m* potential partner agents was selected for reproduction. The population initially contained 1000 individuals (agents), and the genotype of each agent was its set of gathering and transport capacities (evolvable parameters illustrated in *figure 1*). These parameter values directly influenced the heuristic agent lifetime behavior, though the behavioral heuristics (*figure 2*) remained static over the course of the evolutionary process. That is, once an agent had gathered as many mines as it could transport, it would begin transporting the mines back to the home area. During reproduction, agent controller heuristics (*figure 2*) were copied from parent to child, and the fitness inherited by a child was the average fitness of the two parent agents. Ninety percent of the inherited fitness was then subtracted from each parent's fitness.

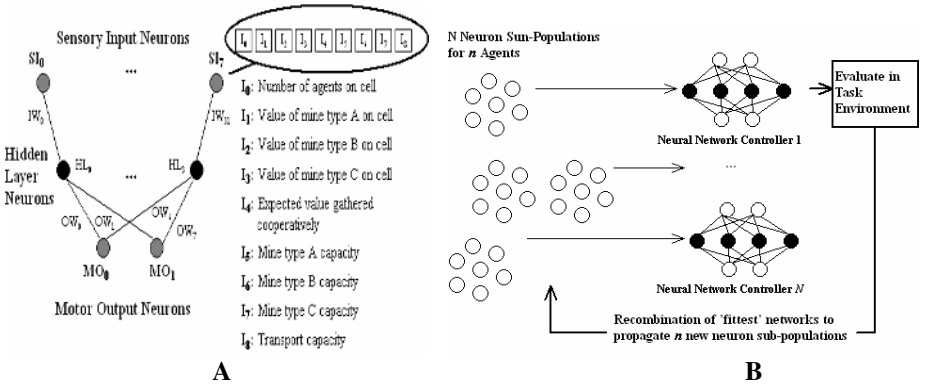


Fig. 3A. Neural network agent controller (neuro-evolution approach). Note that, sensory input neurons SI_1 to SI_6, hidden layer neurons HL_1 and HL_2, input weights IW_1 through to IW_{31}, and output weights OW_2 through to OW_5 are not presented. **B.** Neuro-evolution approach.

2.2 Neuro-evolution Approach

Figure 3A presents the feed-forward neural network agent controller operating under the neuro-evolution approach. Input-output weights connecting the hidden layer neurons from sensory input neurons to motor output neurons were evolved over successive generations under the neuro-evolution process. Agent controller parameter values (the evolvable parameters illustrated in *figure 1*) were evolved over successive generations using a standard evolutionary algorithm [14]. Evolved parameter values were then used as part of the sensory input (*figure 3A*) of the next generation of agents. Thus, as with the pure-evolution approach, the initial population contained 1000 individuals (agents), where the genotype of each agent was the set of input-output weights for the hidden layer of the neural network controller (*figure 3A*), and the set of gathering and transport parameters (*figure 1*). F*igure 3A* presents the neural network controller as having 8 sensory input nodes (SI_0 through to SI_7) to account for 8 surrounding grid-cells, 4 hidden layer nodes (HL_0 through to HL_3), and 2 motor output nodes (MO_0 and MO_1) to account for the *x*, *y* position that the agent moves to. As illustrated in *figure 3A*, each sensory input neuron (SI_0 through to SI_7) was comprised of a 9 value input array. The first four inputs of the array (I_0, I_1, I_2, I_3) correspond to: the *number of agents* observed on the given grid-cell, the value of *mine type A*, *mine type B*, and *mine type C* observed on the given grid-cell, respectively. The fifth value of the sensory input neuron (I_4) was the expected value to be gathered cooperatively. The neural network operated via attempting to select actions that minimized error. Error was the difference between expected value to be gathered cooperatively at simulation time *t*, and actual value gathered cooperatively at simulation time *t+1*. The final four values (I_5, I_6, I_7, I_8) were the *mine types A*, *B*, *C* and *transport* capacities of this agent. The evolvable aspects were the 40 input-output weights connecting hidden layer neurons, and the gathering and transport capacities of the agent.

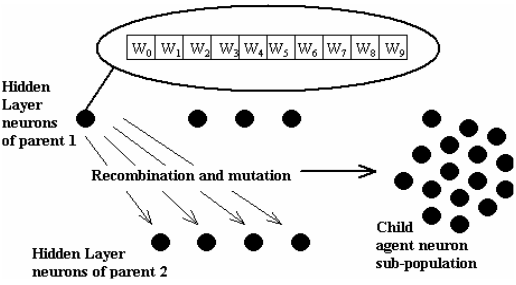


Fig. 4. Neuron reproduction to produce a new child agent sub-population. Note that, only the input-weights of the first hidden layer neuron of parent 1 are presented.

In the neuro-evolution approach, as presented in *figure 3B*, individual neurons for neural network controllers were evolved as a result of being evaluated, and recombined in a social context. As illustrated in *figure 3B*, *n* individual controllers are initially derived by randomly selecting *u* neurons from each sub-population as the *u* neurons for the hidden-layers of *n* controllers. The genetic representation of each sub-population neuron is a string of input and output weights for each hidden-layer neuron.

That is, the approach evolved partial solutions (neurons) that were recombined in novel ways so as to form complete solutions (a group of heterogeneous neural networks). Combinations of hidden layer neurons from two parent agents formed a child sub-population (16 neurons) from which a child network was derived (4 neurons for the hidden layer). *Figure 4* presents the 10 input-output weights of each hidden layer neuron (w_0 to w_9). During reproduction, those in the first parent were recombined (via single-point crossover) and each weight mutated (0.05 probability) with hidden layer neurons in the second parent. This allowed for recombination and mutation of the hidden neuron input-output weights, and produced a new sub-population, from which the fittest 25% of neurons were selected as the hidden layer of a child network.

The key idea of this methodology was that over the course of multiple generations, cooperation occurs within the sub-populations themselves and competition between the *n* sub-populations so as to produce neural network controllers that operate effectively at addressing both individual and social tasks. The key difference delineating this approach from other neuro-evolution methodologies (most notably: ESP [8]) is that it provides a separate sub-population of neurons for the derivation of each neural network. Also, after each evaluation, fitness is assigned to all neurons within a network, and networks can reproduce with any other network in the population of networks. The approach was split into several phases. In the initial phase, the first time the *n* neural network controllers are created, the genotype of each neuron (set of input-output weights) is randomly initialized and *u* neurons are then randomly selected from each of the *n* sub-populations in order to form the hidden-layer of *n* neural network controller. The *n* neural network controllers are then evaluated in the mine sweeping task. Fitness values are awarded to each agent when a mine is delivered to the home area. This fitness is then equally distributed to each hidden-layer neuron participating in the agent's neural network controller.

2.3 Reproduction Schemes and Settings

The experiments compared the four agent reproduction schemes, specifically, SREL and panmictic, SREL and locally restricted, MRDL and panmictic, as well as MRDL and locally restricted reproduction. Each of these schemes was tested and evaluated for a heuristic agent controller with evolvable parameters (operating under a pure-evolution approach), and a neural network controller with evolvable parameters (operating under a neuro-evolution approach).

During the reproduction action, 90% of the fitness of two parent agents was divided amongst and passed onto p offspring agents. During reproduction only one partner agent of m potential partner agents was selected for reproduction. An agent's fitness could only be replenished when it delivered a mine to the home area. The precondition for locally restricted reproduction setting was that there was at least one potential partner agent in the same grid-cell or an adjacent grid-cell. For either the locally restricted or panmictic settings, reproduction was only possible when both parents current fitness was greater than the value of the *min fit reproduction* parameter.

When p offspring agents were produced using the panmictic reproduction setting, each offspring would be placed in a random free grid-cell adjacent to one of the parents. The chance that an offspring an agent was placed in a grid-cell adjacent to parent 1 was 0.5, and the chance that an offspring was placed in a grid-cell adjacent to parent 2 was 0.5. If no adjacent grid cells were free, then the offspring agent died. Using the locally restricted setting offspring agents were always placed in a random free grid-cell adjacent to the parent agent that initiated reproduction. The number of offspring to be produced was determined as m = the total amount of fitness to be inherited (x) divided by 10. According to the reproduction scheme setting being used, pairs of agents produced p offspring using the genetic operations of crossover and mutation [14]. For both the pure evolution and neuro-evolution approaches, the core of reproduction was the application of uniform crossover to 'recombine' the controller parameters: *mine type A, B, C* and *transport capacities* of two parent agents in order to derive the agent controller parameter values of a child agent. The uniform crossover operator selected a parameter value to be inherited from either parent agent with a 0.5 probability. Child controller parameter values were mutated by a value of either plus or minus 10 with a probability of 0.05. If mutation occurred, the probability of adding versus subtracting 10 from the inherited parameter value was 0.5.

3 Experiments, Results, and Discussion

The four agent reproduction schemes were tested and evaluated under the pure-evolution and neuro-evolution approaches. 100 independent runs (each executed for 2000 iterations) were performed. For each of the four reproduction schemes operating under the pure-evolution and neuro-evolution approaches, a control experiment was performed. Each control experiment was non-evolutionary, using static values for the gathering and transport agent controller parameters. The static values utilized were those attained at the end of the evolutionary process (pure-evolution or neuro-evolution) using

a given reproduction scheme. The performance criterion for evolved agent collective behaviors was the *total value of mines gathered cooperatively*.

Table 2 presents the values gathered cooperatively attained for the four reproduction schemes, operating under the pure-evolution and neuro-evolution approaches. For each approach, the values attained in the control experiments are presented below the values gathered cooperatively. The value in parentheses presented next to each of the values gathered cooperatively is the standard deviation. A high standard deviation indicates that the agent collective was less stable in its gathering behavior. High standard deviations were the result of many agent populations (of the 100 replications) becoming extinct before the end of a simulation. A low standard deviation indicates a low portion of agent populations dying out prematurely and hence a high stability in the gathering task. Here, the term *stability* indicates that, for the gathering and transport parameter values evolved, a particular value gathered cooperatively (plus or minus some variance) was expected.

The control experiments demonstrated that both the pure-evolution and neuro-evolution approaches (using the SREL and locally restricted reproduction scheme) were operating within a region of the parameter space (defined by the four agent controller parameters) where a high value gathered cooperatively was attainable. This was especially the case for the neuro-evolution approach, which, when using the SREL and locally restricted, and SREL and panmictic reproduction schemes, was able to attain values gathered cooperatively over an order of a magnitude higher than comparative agent collectives.

Also, *table 2* highlights that, agents using the SREL and panmictic reproduction scheme and operating under the neuro-evolution approach, were able to achieve a higher stability comparative to the other reproduction schemes. This is theorized to be a result of the panmictic reproduction scheme that selects a partner agent from anywhere in the environment.

Under the neuro-evolution approach, panmictic reproduction encourages and preserves the heterogeneity and diversity in the n sub-populations corresponding to the n agent controllers. Locally restricted reproduction restricts the diversity produced in child sub-populations (hence agent controllers) by only selecting from agent sub-populations local to the proximity of the reproducing agent.

Under the pure-evolution approach, all agent controllers are initialized with the same heuristics, and the agent controllers do not evolve over successive generations. This heterogeneity of controllers under the neuro-evolution approach, and the homogeneity of controllers under the pure-evolution approach, refers only to the structure of the agent controllers, and not to the evolvable parameters (as used in both approaches).

The result of the SREL and locally restricted agent reproduction scheme being most appropriate for both approaches (pure-evolution and neuro-evolution) is theorized to be consequent of agents only reproducing at the end of their lifetimes. Using the SREL setting, agents that have performed their task well and have thus survived until the end of allotted lifetime, are allowed reproduce. Given that the reproduction action costs 90% of the parents' energy, agents using the MRDL setting have less of a chance of producing offspring that are well suited to successful task accomplishment.

Table 2. The values attained for the total value gathered cooperatively (standard deviations in parentheses) under pure-evolution and neuro-evolution. Values attained in the control experiments are presented under the respective approach and reproduction scheme setting used.

	SREL Panmictic	**SREL Local**	**MRDL Panmictic**	**MRDL Local**
Neuro-Evolution	**159.67** (12.96)	**300.95** (46.56)	**37.92** (7.75)	**30.61** (4.90)
Control	**610.23** (9.10)	**870.67** (60.34)	**92.91** (3.67)	**68.50** (2.93)
Evolution	**23.59** (33.37)	**39.10** (17.20)	**32.56** (10.00)	**22.85** (17.60)
Control	**60.25** (1.85)	**71.70** (3.50)	**43.04** (5.46)	**54.28** (0.63)

This is especially the case for the neural-evolution approach, since neural network controller weights need sufficient time to change and produce an effective agent behavior, in order for that behavior to be propagated in the next generation of agents. In the case of the heuristic controller, child agents inherit only recombined and mutated agent parameter values and an average of parent fitness. However, the nature of the SREL setting holds, in that only agents with appropriate controller parameter settings will have survived until the end of their allotted lifetime (that is, those agents with a high fitness).

Hence, *table 2* illustrates that for the pure-evolution and neuro-evolution approaches, the SREL and locally restricted reproduction scheme is the most appropriate for the given task. It is theorized the superior performance of the neuro-evolution approach is a result of agent lifetime behavior adapting over successive generations, and no direct reliance upon controller parameter values. The heuristic behavior under the pure-evolution approach relies directly upon the values of the gathering and transport capacities in order for an agent to decide where to move and what type of mine can be gathered.

Furthermore, as a benchmark to illustrate the efficacy of evolved agent controller parameter values, additional control experiments were run using the four reproduction schemes under the pure-evolution and neuro-evolution approaches. These control experiments utilized the maximum possible values (at initialization) for the gathering and transport parameters. That is, 100, 100, 100, and 300 for the *mine type A*, *B*, *C*, and *transport* capacities respectively.

The resulting values gathered cooperatively (average taken over 100 runs) were always low with high standard deviations (comparative to values attained in other experiments) for collectives using the pure-evolution approach. The low values and high standard deviations for each of the reproduction scheme settings operating under the pure-evolution approach indicate that all agent populations died prematurely.

Under the neuro-evolution approach, low values gathered cooperatively and high standard deviations were attained, indicative of few collectives (of the 100 replications) surviving until the final simulation iteration. This was a result of high values for the agent controller gathering and transport capacities (*table 1*) yielding correspondingly high gathering and transport costs, where these costs usually exceeded an agent's fitness.

4 Conclusions

This paper compared the efficacy of different agent reproduction scheme settings for accomplishing a cooperative gathering task. Results indicated that agent collectives utilizing the single reproduction at end of lifetime (SREL) and the locally restricted reproduction scheme yielded a superior performance in a collective gathering task. This agent reproduction scheme setting attained the highest performance in terms of the evaluation criterion for both a heuristic agent controller (operating under a pure-evolution approach) and a neural network agent controller (operating under a neuro-evolution approach). The evaluation criterion was defined as the total value of resources gathered cooperatively in a simulated environment within a given time period.

References

1. Avouris, N., Gasser, L.: Distributed Artificial Intelligence: Theory and Praxis. Kluwer Academic Publishers. Dordrecht (1992).
2. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, Oxford (1998).
3. Hertz, J., Krogh, A., & Palmer, R.: Introduction to the Theory of Neural Computation. Addison-Wesley. Redwood City (1991).
4. Gomez, F., & Miikkulainen, R (1997).: Incremental evolution of complex general behavior. *Adaptive Behavior*, vol. 5(1): 317-342.
5. Nolfi, S., Baldassarre, G., & Parisi, D.: Evolution of collective behavior in a team of physically linked robots. In Gunther, R., Guillot, A, Meyer, J. (Eds.), *Applications of Evolutionary Computing* (pp. 581-592). Springer-Verlag: Heidelberg (2003).
6. Michel, O.: Evolutionary Neuro-genesis Applied to Mobile Robotics. In, Honovar, V. (Ed.). *Advances in Evolutionary Synthesis of Neural Systems*. MIT Press: Cambridge (1999).
7. Floreano, D. & Urzelai, J. (2000). Evolutionary Robots with on-line self-organization and behavioral fitness. *Neural Networks*, vol. 13(1): 431-443.
8. Gomez, F. Robust Non-Linear Control through Neuro-evolution. PhD Thesis. Department of Computer Sciences. University of Texas: Austin (2003).
9. Gomez, F., Miikkulainen, R. Solving Non-Markovian Control Tasks with Neuro-evolution. In Dean, T. (Ed.), *Proceedings of the International Joint Conference on Artificial Intelligence*, (pp. 1356-1361). Morgan Kaufmann: Stockholm (1999).
10. Moriarty, D., Miikkulainen, R. (1997). Forming Neural Networks through Efficient and Adaptive Co-evolution. *Evolutionary Computation*, 5(4): 373-399.
11. Whiteson, S., Kohl, N., Miikkulainen, R., Stone, P. Evolving Keep-away Soccer Players through Task Decomposition. In *Proceeding of the Genetic and Evolutionary Computation Conference*, (pp. 356-368). Springer-Verlag: Berlin (2003).
12. Yong, C., & Miikkulainen, R.: Cooperative Co-evolution of Multi-Agent Systems. *Technical Report*. Computer Sciences Department, University of Texas. Austin, USA (2001).
13. Bryant, B., Miikkulainen, R.: Neuro-evolution for Adaptive Teams. In *Proceedings of the 2003 Congress on Evolutionary Computation*, (pp. 2194-2201). IEEE Press: Canberra (2003).
14. Eiben, A. E. & Smith, J. E.: Introduction to Evolutionary Computing, Springer-Verlag: Berlin (2003).

Construction-Based and Inspection-Based Universal Self-replication

André Stauffer, Daniel Mange, and Gianluca Tempesti

Ecole Polytechnique Fédérale de Lausanne (EPFL),
Logic Systems Laboratory, CH-1015 Lausanne, Switzerland
andre.stauffer@epfl.ch, lslwww.epfl.ch.

Abstract. After a survey of some typical realizations of self-replicating machines, this paper presents the self-replication based on construction and the self-replication based on inspection of an interactive loop, chosen as an easily understandable example. The construction-based replication process is achieved by translation and transcription of the configuration information of the loop in the processing unit of a data and signals cellular automaton (DSCA). The inspection-based replication process is realized by duplication and translation of the same configuration information in the processing unit of the DSCA.

1 Introduction

In the history of non trivial self-replicating machines, there are mainly two different approaches: (1) the self-replication based on construction, and (2) the self-replication based on inspection.

Using *construction* in order to self-replicate structures was the way von Neumann [1] conceived to solve the problem. It was Langton [2] who realized the first practical implementation of the process. In these approaches, the information is successively used in two different modes, interpreted and uninterpreted. First, in the interpreted mode or translation, the signals are executed as they reach the periphery in order to construct the replicated structure. After, in the uninterpreted mode or transcription, the signals are copied in the replicated structure.

Using *inspection* in order to implement self-replicating structures was the approach chosen by a few researchers. Morita and Imai [3] present configurations named worms and loops able to self-replicate by using a shape-encoding method. Ibanez and al. [4] describe also a self-inspecting loop capable of self-replication. In these works, the data are only used in an uninterpreted way called transcription. It is accomplished by duplication of signals at the periphery of the structure.

The main goal of this paper is to carry out construction and inspection in order to self-replicate a simple interactive loop [5]. These self-replication processes are achieved in data and signals cellular automata (DSCA) [6] implementing the so-called Tom Thumb algorithm which allows the design of

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 805–814, 2005.

structures with universal construction properties [7]. The fundamentals of this algorithm and its application to the DSCA implementation of construction-based loop replication are defined in Section 2. Section 3 presents the modified Tom Thumb algorithm and its application to the DSCA implementation of the inspection-based loop replication. Section 4 will conclude by comparing functionally and structurally the DSCA implementations of the two self-replicating processes.

2 Construction-Based Replication

Using the Tom Thumb algorithm [7] in order to configure a 4×3 interactive loop (Figure 1a), chosen as an easily understandable example, the string of data given in Figure 1b is applied twice.

This configuration string, which consists of alternate structural data (Figure 1c) and functional data (1, 2, ..., 10), allows the growth of the loop every four time-steps (Figure 2a). The corresponding cellular signals and cellular modes are defined in Figure 2b and 2c.

An external input, applied to one of the four cells having a branch data (i.e. the corner cells), initiates the construction-based self-replication of the loop. According to the branch data of the cell, this process generates first an east, north, west or south directed growth signal. It provides then twice the string of data in order to construct the loop structure (translation) and to save a copy of the configuration in the replicated loop (transcription).

We will now describe the detailed architecture of our basic cell which corresponds to a data and signals cellular automaton (DSCA) cell [6]. This DSCA cell is designed as a digital system, resulting from the interconnection of a processing unit handling the data and a control unit computing the signals.

The processing unit is made up of three resources (Figure 3):

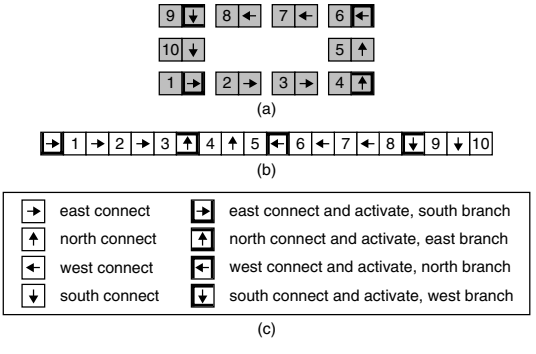


Fig. 1. Interactive loop. (a) Cellular structure. (b) Configuration string. (c) Structural data.

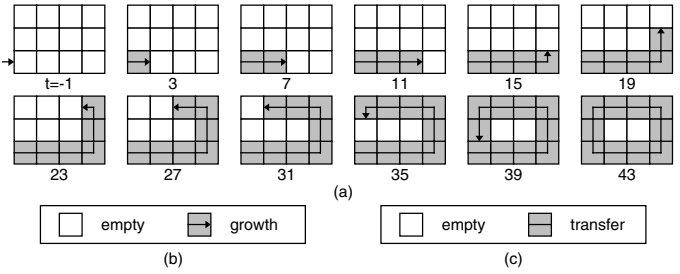


Fig. 2. Growth of the loop. (a) Automaton evolution. (b) Cellular signals. (c) Cellular modes.

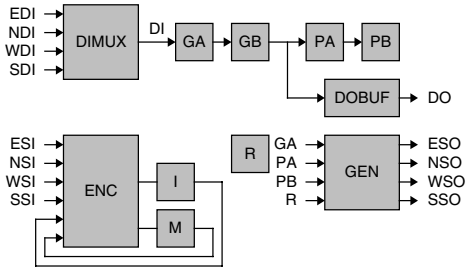


Fig. 3. Detailed architecture of the DSCA cell

- An input multiplexer DIMUX, selecting one out of the four input data EDI, NDI, WDI or SDI.
- A 4-level stack organized as two registers GA and GB (for mobile string data), and two registers PA and PB (for fixed string data).
- An output buffer DOBUF producing the output data DO.

The control unit consists of five resources (Figure 3):

- An encoder ENC for the input signals ESI, NSI, WSI, and SSI.
- A transmission register I for the memorization of the input selection.
- A mode register M.
- A replication register R.
- A generator GEN producing the output signals ESO, NSO, WSO, and SSO.

The specifications of these resources are given in Appendix A.

3 Inspection-Based Replication

In the inspection-based replication process, only one string of data is needed to construct the replicated structure. This string is obtained by duplication of the configuration data of the structure to replicate (transcription).

To replicate the 4×3 interactive loop (Figure 1a), the Tom Thumb algorithm [7] must be modified in order to recreate the configuration string (Figure 1b) from the structure of the loop. The increased complexity of the process involves new cellular signals and cellular modes (Figure 4).

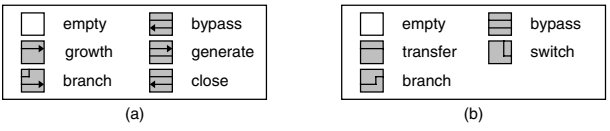


Fig. 4. Replication process. (a) Cellular signals. (b) Cellular modes.

Performed on one of the four cells having a branch data (i.e. the corner cells), an external input initiates the inspection-based self-replication of the loop. When the external input is applied to the lower right cell of the loop at time-step t =i, the replication process builds first the bypass datapath of Figure 5.

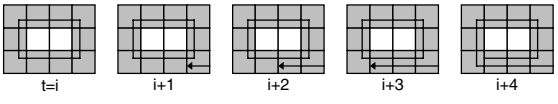


Fig. 5. Bypass datapath construction

Starting from the lower right cell, the process recreates then the configuration string by selecting successively the fixed data (structural data and functional data) of each cell and propagating them as mobile data through a transmission register (Figure 6).

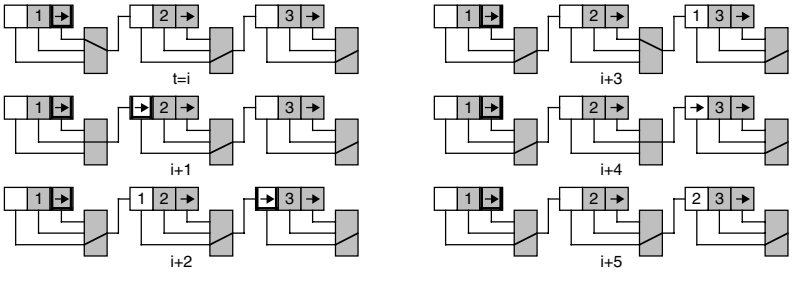


Fig. 6. Configuration string generation

As the recreated string reaches the lower right cell, it starts the growth of the replicated loop every three time-steps (Figure 7).

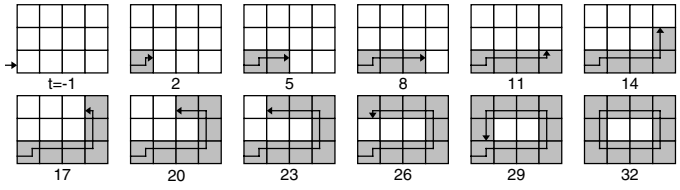


Fig. 7. Growth of the loop

Finally, when the replication is accomplished, the bypass datapath of the replicating loop is suppressed (Figure 8).

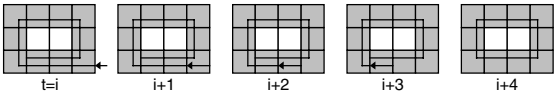


Fig. 8. Bypass datapath suppression

In order to carry out the whole process needed for the inspection-based replication of the loop, the implementation of the DSCA cell comprises a total of 15 resources. The processing unit interconnects eight of them (Figure 9):

- Two input multiplexers D1IMUX (resp. D2IMUX), selecting one out of the four input data $ED1I$, $ND1I$, $WD1I$ or $SD1I$ (resp. $ED2I$, $ND2I$, $WD2I$ or $SD2I$).
- Two switch multiplexers D1MUX (resp. D2MUX), selecting the outputs of the input multiplexers.
- A 3-level stack organized as a transfer register GA (for mobile data), and two configuration registers PA and PB (for fixed data).
- A bypass register GB.
- An output multiplexer DOMUX selecting the output data $D1O$.
- An output buffer DOBUF producing the output data $D2O$.

The control unit involves the seven remaining ones (Figure 9):

- An encoder ENC for the input signals ESI, NSI, WSI, and SSI.
- A transmission register I for the memorization of the input selection.

- A signal register S.
- A mode register M.
- An output register O to control the output data $D1O$.
- A replication register R.
- A generator GEN producing the output signals ESO, NSO, WSO, and SSO.

The specifications of these resources are given in Appendix B.

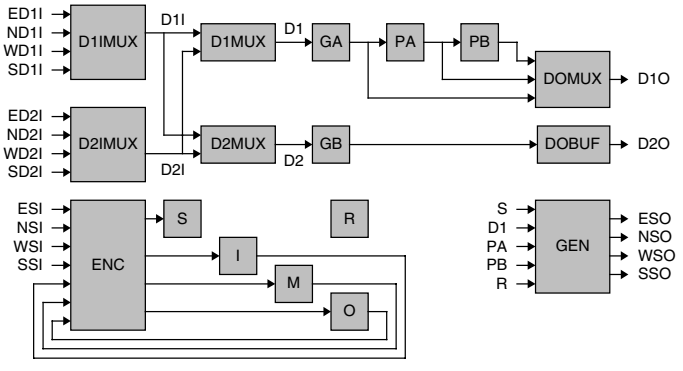


Fig. 9. Detailed architecture of the DSCA cell

4 Conclusion

In our construction-based approach, a copy of the configuration string is always circulating and the replication process consists to pick this information twice. First, in order to construct the replicated structure and after, in order to supply the circulating information.

In our inspection-based approach, there is no moving information. A copy of the configuration string is obtained by duplication of the fixed data. This copy is sufficient for the construction of the replicated structure, but the replication process in order to get it is more complex.

For the loop example, where the functional data are minimal (one digit per cell), the hardware cost of the inspection-based implementation exceeds by far that of the construction-based one. However, for a less trivial example with a greater amount of functional data, the hardware costs of the implementations are just the opposite.

The construction-based and inspection-based loops presented in this paper are implemented in our reconfigurable electronic medium for bio-inspired systems: the BioWall (Figure 10). This medium is a vertical board made of 2000 units with input, output and computation abilities [8].

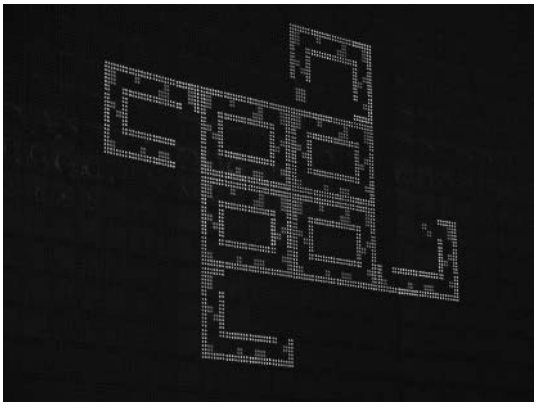


Fig. 10. The BioWall used to physically implement the loops

References

1. J. von Neumann. Theory of Self-Reproducing Automata. University of Illinois Press, Illinois, 1966. Edited and completed by A. W. Burks.
2. C. Langton. Self-reproduction in cellular automata. Physica D, vol.10, pp.135–144, 1984.
3. K. Morita and K. Imai. Logical universality and self-reproduction in reversible cellular automata. In T. Higuchi, M. Iwata, and W. Liu (Eds.), Proceedings of The First International Conference on Evolvable Systems: From Biology to Hardware (ICES96). Lecture Notes in Computer Science, pp.152–166, Springer-Verlag, Heidelberg, 1997.
4. J. Ibanez, D. Anabitarte, I. Azpeitia, O. Barrera, A. Barrutieta, H. Blanco, and F. Echarte. Self-inspection based reproduction in cellular automata. In A. Moreno, J.J. Merelo, and P. Chacon (Eds.), Proceedings of the European Conference on Artificial Life (ECAL95). Advances in Artificial Life, Springer-Verlag, Heidelberg, 1995.
5. A. Stauffer and M. Sipper. An interactive self-replicator implemented in hardware. Artificial Life, vol.8, no.2, pp.175–183, 2002.
6. A. Stauffer and M. Sipper. The data-and-signals cellular automaton and its application to growing structures. Artificial Life, vol.10, no.4, pp.463–477, 2004.
7. D. Mange, A. Stauffer, E. Petraglio, and G. Tempesti. Self-replicating loop with universal construction. Physica D, vol.191, no.1-2, pp.178–192, 2004.
8. G. Tempesti, D. Mange, A. Stauffer, and C. Teuscher. The BioWall: An elctronic tissue for prototyping bio-inspired systems. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. S. Zebulum (Eds.), Proceedings of the 2002 NASA/DOD Workshop Conference on Evolvable Hardware, pp.221–230, IEEE Computer Society Press, Los Alamitos CA, 2002.

Appendix A: Construction-Based Specification

Figures 11 and 12 are the truth tables and transition tables of the resources needed for the implementation of the construction-based replication cell.

M	I	DI
empty	-	empty
transfer	sele	EDI
transfer	seln	NDI
transfer	selw	WDI
transfer	sels	SDI

(a)

PB	DO
empty	empty
empty'	GB

(b)

GA	PA	PB	R	ESO	NSO	WSO	SSO
-	east connect	empty	-	growth	0	0	0
-	south branch	empty	-	growth	0	0	0
south branch	-	east branch	1	growth	0	0	0
-	north connect	empty	-	0	growth	0	0
-	east branch	empty	-	0	growth	0	0
east branch	-	north branch	1	0	growth	0	0
-	west connect	empty	-	0	0	growth	0
-	north branch	empty	-	0	0	growth	0
north branch	-	west branch	1	0	0	growth	0
-	south connect	empty	-	0	0	0	growth
-	west branch	empty	-	0	0	0	growth
west branch	-	south branch	1	0	0	0	growth

(c)

Fig. 11. Truth tables. (a) Input multiplexer DIMUX. (b) Output buffer DOBUF. (c) Output generator GEN (0: empty signal).

PB	GA+	GB+	PA+	PB+
empty	DI	GA	GB	PA
empty'	DI	GA	PA	PB

(a)

R	EXT	PB	S0	R+
off	input	branch	-	on
on	-	-	growth	off

(b)

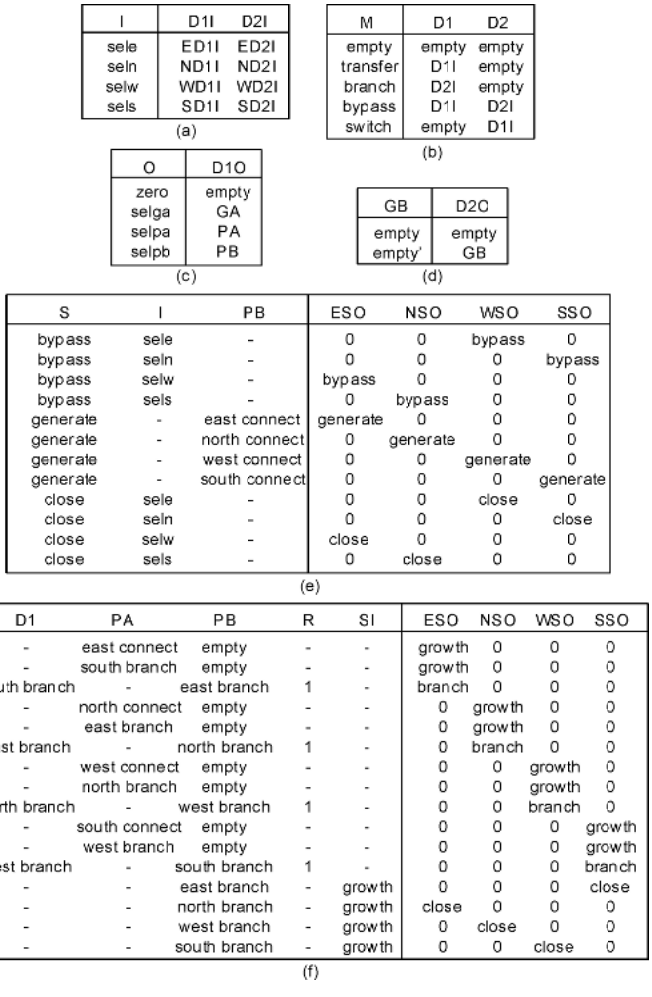
M	I	ESI	NSI	WSI	SSI	M+	I+
empty	-	growth	-	-	-	transfer	sele
empty	-	-	growth	-	-	transfer	seln
empty	-	-	-	growth	-	transfer	selw
empty	-	-	-	-	growth	transfer	sels
transfer	sele	-	-	-	growth	transfer	sels
transfer	seln	growth	-	-	-	transfer	sele
transfer	selw	-	growth	-	-	transfer	seln
transfer	sels	-	-	growth	-	transfer	selw

(c)

Fig. 12. Transition tables. (a) 4-level stack. (b) Replication register R. (c) Input encoder ENC.

Appendix B: Inspection-Based Specification

Figures 13 and 14 are the truth tables and transition tables of the resources needed for the implementation of the inspection-based replication cell.



I	D1I	D2I
sele	ED1I	ED2I
seln	ND1I	ND2I
selw	WD1I	WD2I
sels	SD1I	SD2I

(a)

M	D1	D2
empty	empty	empty
transfer	D1I	empty
branch	D2I	empty
bypass	D1I	D2I
switch	empty	D1I

(b)

O	D1O
zero	empty
selga	GA
selpa	PA
selpb	PB

(c)

GB	D2O
empty	empty
empty'	GB

(d)

S	I	PB	ESO	NSO	WSO	SSO
bypass	sele	-	0	0	bypass	0
bypass	seln	-	0	0	0	bypass
bypass	selw	-	bypass	0	0	0
bypass	sels	-	0	bypass	0	0
generate	-	east connect	generate	0	0	0
generate	-	north connect	0	generate	0	0
generate	-	west connect	0	0	generate	0
generate	-	south connect	0	0	0	generate
close	sele	-	0	0	close	0
close	seln	-	0	0	0	close
close	selw	-	close	0	0	0
close	sels	-	0	close	0	0

(e)

D1	PA	PB	R	SI	ESO	NSO	WSO	SSO
-	east connect	empty	-	-	growth	0	0	0
-	south branch	empty	-	-	growth	0	0	0
south branch	-	east branch	1	-	branch	0	0	0
-	north connect	empty	-	-	0	growth	0	0
-	east branch	empty	-	-	0	growth	0	0
east branch	-	north branch	1	-	0	branch	0	0
-	west connect	empty	-	-	0	0	growth	0
-	north branch	empty	-	-	0	0	growth	0
north branch	-	west branch	1	-	0	0	branch	0
-	south connect	empty	-	-	0	0	0	growth
-	west branch	empty	-	-	0	0	0	growth
west branch	-	south branch	1	-	0	0	0	branch
-	-	east branch	-	growth	0	0	0	close
-	-	north branch	-	growth	close	0	0	0
-	-	west branch	-	growth	0	close	0	0
-	-	south branch	-	growth	0	0	close	0

(f)

Fig. 13. Truth tables. (a) Selection multiplexers D1IMUX and D2IMUX. (b) Switch multiplexers D1MUX and D2MUX. (c) Output multiplexer DOMUX. (d) Output buffer DOBUF. (e) Output generator GEN (growth, branch and close signals; 0: empty signal). (f) Output generator GEN (bypass, generate and close signals).

PB	GA+	PA+	PB+	GB+
empty	D1	GA	PA	D2
empty'	D1	PA	PB	D2

(a)

R	EXT	PB	SO	R+
off	input	branch	-	on
on	-	-	growth	off

(b)

M	ESI	NSI	WSI	SSI	I+
empty	growth	-	-	-	sele
empty	-	growth	-	-	seln
empty	-	-	growth	-	selw
empty	-	-	-	growth	sels
empty	branch	-	-	-	sele
empty	-	branch	-	-	seln
empty	-	-	branch	-	selw
empty	-	-	-	branch	sels
branch	growth	-	-	-	sele
branch	-	growth	-	-	seln
branch	-	-	growth	-	selw
branch	-	-	-	growth	sels

(c)

M	SI	PB	R	M+	S+
empty	growth	-	-	transfer	0
empty	branch	-	-	branch	0
transfer	-	-	1	bypass	bypass
transfer	bypass	connect	-	bypass	bypass
transfer	bypass	branch	-	switch	0
branch	growth	-	-	transfer	0
bypass	close	-	-	transfer	close
switch	close	-	-	transfer	0

(d)

O	M	SI	PA	PB	O+	S+
zero	-	-	empty'	-	selga	0
selga	transfer	bypass	-	branch	selpb	0
selga	transfer	generate	-	-	selpb	0
selga	bypass	generate	-	-	selpb	0
selpb	-	-	-	-	selpa	0
selpa	-	-	-	-	selga	generate

(e)

Fig. 14. Transition tables. (a) 3-level stack. (b) Replication register R. (c) Input encoder ENC (input data control). (d) Input encoder ENC (cell mode and signal). (e) Input encoder ENC (output data control).

Coordinating Dual-Mode Biomimetic Robotic Fish in Box-Pushing Task

Dandan Zhang, Yimin Fang, Guangming Xie, Junzhi Yu, and Long Wang

Intelligent Control Laboratory, Center for Systems and Control,
Department of Mechanics and Engineering Science,
Peking University, Beijing 100871, P. R. China
twinkleice@pku.edu.cn

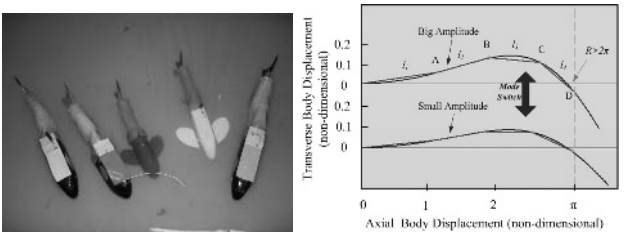
Abstract. This paper presents a novel method for coordinating multiple biomimetic robotic fish in box-pushing task. Based on our successfully developing a robotic fish prototype of which the swimming modes can be switched flexibly and smoothly, we step further to study coordination problems of multiple robotic fish in unstructured and dynamic environments. To simplify the difficulties of path planning and action decision when the robotic fish is approaching the box, we employ the *situated-behavior* method, and for each situation a specific behavior is elaborately designed. On dealing with the synchronization and coordinated pushing problems in the particular underwater environment, fuzzy logic method is adopted for motion planning of the fish. Experimental results of box-pushing performed by two robotic fish validate the effectiveness of the proposed method.

1 Introduction

Fish, through billions of years evolution, has long been the unique king of the ocean. Natural selection ensures that the mechanical systems involved in fish are efficient and adaptive to their living environments. It is well-known that a fish in nature propels itself by coordinated motion of the body, fins, and tail, achieving tremendous propulsive efficiency and excellent maneuverability. From the perspective of engineering science, fish is a prototype of a distinguished autonomous underwater vehicle. Taking advantage of progress in robotics, control technology, artificial intelligence, hydrodynamics of fish-like swimming, new materials, sensors, and actuators, emerging research has been focused on developing novel fish-like vehicles, to imitate the locomotion mechanism of fish in nature to get favorable efficiency, maneuverability and low noise performance. Research on robotic fish has become a hot topic and received much attention[1]-[6]. Observations show that a majority of fishes in nature are gregarious. As a colony, fish can exhibit incredible power on foraging for foods, avoiding enemies/predators, cruising, and so on. Since most of the previous research on robotic fish was mainly concerned with the hydrodynamic mechanism of fishlike swimming, coordinated control of multiple robotic fish is definitely an important research topic for future engineering applications. However, at present, few research results are available in the literature. In addition, although fruitful cooperation methods and valuable experimental results for ground robots have been published, it's very difficult to apply these methods directly

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 815–824, 2005.

to robotic fish due to the complexity and particularity of the underwater working environment and the particular movement modes of robotic fish. Coordinated control of multiple robotic fish remains certainly a challenging research topic.



(a) The Fishes (b) Links Based Body-wave Fitting

Fig. 1. The Fishes and Links Based Body-wave Fitting

Previously we have developed several radio-controlled, four-linked robotic fish of different forms and functionalities (shown in Fig.1(a)), each propelled by the flexible posterior body and the oscillatory tail. In this paper, we study multiple robotic fish coordination problem in context of the box-pushing task, since box-pushing has long been viewed as one of the canonical task domains and benchmark problems for cooperative robotics[7]. In the box-pushing task, several robotic fish are required to cooperatively move a rectangular box from an initial location to a designated goal location in the water environment. Because of the complex hydrodynamics of the fluid environment and dynamics of the fish when it swims, it is difficult to establish precise mathematical model by purely analytical methods[1], and we can only predict approximately the response of the fish on the control commands. Under these limitations, to simplify the path planning and action decision for the individual fish when it is in different positions relative to the box, we utilize the *situated-behavior* method to divide the environment into a set of complete and exclusive situations, and for each situation, a specific behavior is elaborately designed, which may include several actions. To implement the method, a geometry-based approach called Comfortable Circle Approach (CCA) is proposed. Since the robotic fish can not stop immediately on a ”stop” command, fuzzy logic method is employed to facilitate the synchronous arrival of the two fish to the box. The direction of the box is controlled by the coordinated motion of the fish, obtained through a set of fuzzy logic rules. In fact, when a robotic fish is in different situations, different motion performances are required, i.e. when the robotic fish is far from the box, the fish shall swim with high speed toward the box to reduce the time consuming; while when the fish has come close to the box, precise point-to-point control is required to lead the fish to the Attacking Point (on the box, where the fish pushes the box), and the fish shall swim smoothly. However, fast swimming performance and smooth swimming performance conflict with each other. We define two different motion modes for the robotic fish, FAST-SWIMMING mode and SMOOTH-SWIMMING mode. So the robotic fish is called dual-mode fish, and the mode switch is embedded in our control method. Experiments on two robotic fish validate the effectiveness of our proposed method.

The paper is organized as follows. In section 2, a review of the robotic fish design is presented. In section 3, we describe our box-pushing task and the coordination method in detail. Experimental results are given in section 4. Section 5 concludes the paper.

2 Robotic Fish Prototype

We have designed and implemented several radio-controlled, 4-link and free-swimming biomimetic robotic fish. Each fish uses a flexible posterior body and an oscillating foil as the propulsor. Next, we give a brief review of the prototype of the robotic fish.

2.1 Simplified Propulsive Model of the Robotic Fish

Body and/or caudal fin (BCF) swimming movements of natural fish are usually categorized into anguilliform, carangiform, subcarangiform, and thunniform mode primarily according to the wavelength and the amplitude envelope of the propulsive wave underlying fish′s behavior[1][8]. Recent research on robotic fish mainly focuses on the anguilliform swimming mode and the carangiform swimming mode. Carangiform swimmers are generally faster than anguilliform swimmers, and also the carangiform propulsion is more convenient for engineering implementation. For carangiform movement, the undulation of the swimmer's body is mainly confined to the last 1/3 part of the body. Barrett et al. has presented a relative swimming model for RoboTuna (carangiform) in [9], and the undulatory motion is assumed to take the form of a propulsive travelling wave. The discrete form of the wave is described by:

$$y_{body}(x,i) = [(c_1 x + c_2 x^2)][\sin(kx - \frac{2\pi}{M} i)], \tag{1}$$

where y_{body} is the transverse displacement of the fish body, x the displacement along the main axis, k the body wave number, i the index of the sequences in an oscillation cycle, M the body wave resolution, representing the discrete degree of the overall travelling wave. The designed oscillatory part of a robotic fish consists of several rotating hinge joints, as shown in Fig. 1(b). It is modelled as a planar serial chain of links along the axial body displacement, and the positions of the links in the chain is achieved by numerical fitting. See [1] for details of determination and optimization of the ratio $l_1 : l_2 : ... : l_n$. The swimming mode of the robotic fish can be switched flexibly between the FAST-SWIMMING (with big oscillating amplitude) mode and the SMOOTH-SWIMMING (with small oscillating amplitude) mode.

3 Multiple Robotic Fish Box-Pushing

3.1 Task Description

The box-pushing task is stated as follows: Two robotic fish are required to coordinately move a rectangular box 365 mm $\times$260 mm $\times$95 mm (length $\times$ width $\times$ depth) from an initial location to a goal location in a swim pool 3200 mm $\times$2200 mm $\times$700 mm (length $\times$ width $\times$ depth). The box has a direction and the fish are only allowed to push

the box behind the box with their heads, one on the left side, the other on the right side. The directions of the box and the fish can be recognized by the overhead camera. In fact, the locomotion of a robotic fish is nonlinear, and the changing swimming pattern makes it difficult for the fish to keep balance. Difficulties of robotic fish control in the box-pushing task are briefly summarized as follows:

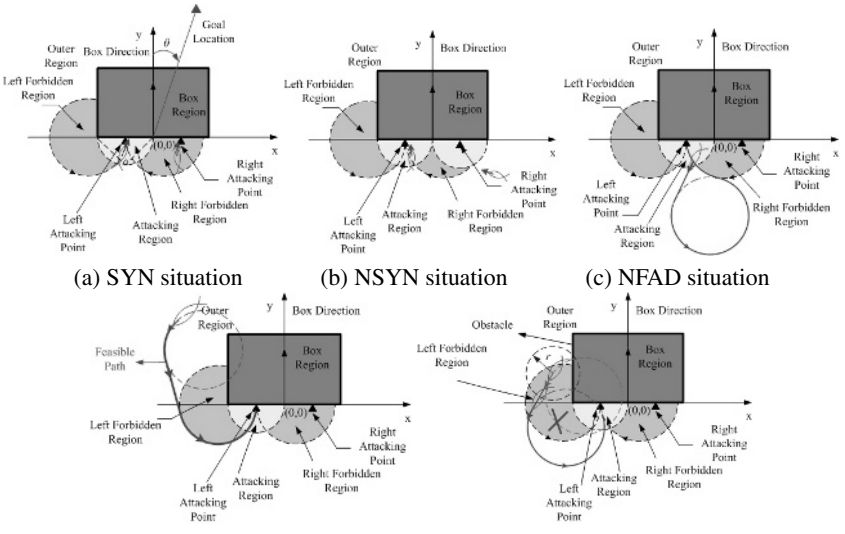
- ⋆ It is difficult for a robotic fish to trace a linear path. In addition, the fish can not back off like a wheeled ground robot.
- ⋆ The robotic fish can not stop immediately on receiving a "stop" command because the resistance in water is much less than that on the ground. So the two fish shall synchronize the arrival at the Attacking Points of the box, but it is hard to control.
- ⋆ Waves occur when a robotic fish moves, which affect the movement of the robotic fish and others even in static state. This leads to great difficulty in precise point-to-point (PTP) control. Also the box will drift and rotate when waves occur, or collided by the fish.
- ⋆ Since it is very difficult to establish the hydrodynamic model of fishlike swimming using analytical method, we can only predict approximately the response of the robotic fish on a control command.

Under the these limitations, the box-pushing problem is decomposed into two sub-problems: path planning, which is to plan a feasible path allowing the fish to reach the Attacking Point under the limitation of the minimum turn radius; and motion planning, which is to lead the two fish to arrive at the Attacking Points synchronously, and to control the direction of the box toward the goal location by coordinating the swimming speeds of the two fish.

3.2 Comfortable Circle Approach to Path Planning

Before the fish can push the box, they shall first reach the box, so a feasibl e path shall be planned for the fish to trace. Considering the difficulties in precise locomotion control mentioned above, we employ the "situated-behavior" method to reduce the difficulties of path planning and action casting in the box-pushing task. This method divides the environment into a set of *exclusive* and *complete* situations, and for each situation, a behavior is elaborately designed to solve the situation associated problem individually. The advantage of employing this method is that it is a "divide and conquer" strategy, which reduces the task difficulty; in addition, the real-time *behavior coordination problem* need not to be taken into consideration, since the situations are complete and exclusive. These behaviors can also include and share low-level behaviors, i.e., obstacle-avoidance behavior. We propose a geometry-based implementation of the *situated-behavior* method called Comfortable Circle Approach (CCA). Next we will describe CCA in detail.

The Situations. The situations are obtained according to the relative states of the problem entities (i.e. the robotic fish, the box, the Attacking Points, and the goal location). Shown in Fig. 2, the environment is divided into Box Region, Attacking Regions, Left Forbidden Region, Right Forbidden Region, and Outer Region. The Attacking Regions are semicircular regions centered at the Attacking Points with radius of $1/4$ length



(a) SYN situation (b) NSYN situation (c) NFAD situation

(d) FP situation (e) NFP situation

Fig. 2. Situations and Associated Behaviors Design

of the box. In these regions, the fish can attack the box directly. The boundaries of the Forbidden regions are directed circles with constant radius, passing the Attacking Points with directions the box direction. We assume that the radius is 1.3 length of the minimum turn radius of the fish and call it a "comfortable" radius which means that the fish can turn comfortably with this radius at each speed. A directed circle with a "comfortable" radius is called a *Comfortable Circle*. Here we discuss only the situations for the Left Attacking Point and the fish whose goal attacking point is the left one. The situations for the Right Attacking Point and the fish whose attacking point is the right one are similar. Our objective is to plan appropriate paths which lead the robotic fish to arrive at the Attacking Points with the box direction, in order to prepare for the next step − box-pushing. We use a decision tree to define the set of situations according to the relative states of the problem entities. The decision tree is traversed through binary decision rules according to several criteria (the situations are hand-designed, however, with different criteria, different situations can be defined, and these situations are exclusive and complete). As shown in Fig. 3, the inputs of the decision tree are the goal location information and sensory information from the overhead camera, including the locations and directions of the fish and the box. The current situation is identified according to the input information. Finally we derive five complete and exclusive situations:

(1) SYN (synchronizing) situation: both fish are in the Attacking Regions and their directions are feasible, that is, the directions are consistent with the box direction within a constant limit (in our experiment, we select 45°), as well intersect with the boundary of the box)(see Fig. 2(a)).

(2) NSYN (not synchronizing) situation: the other fish is not in the Attacking region or its direction is not feasible (see Fig. 2(b)).

(3) NFAD (not feasible attacking direction) situation: The fish is in the Attacking Region but not with a feasible direction (see Fig. 2(c));

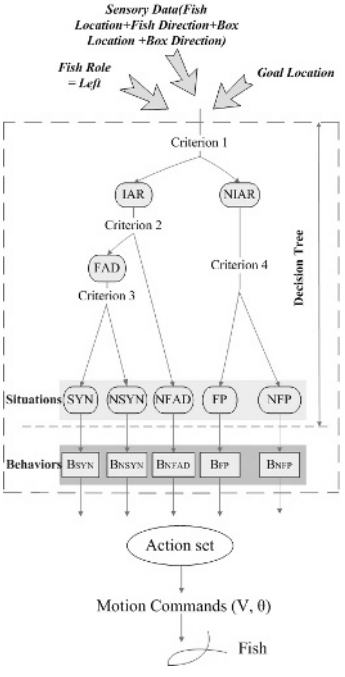


Fig. 3. Decision Tree of CCA

(4) FP (Feasible Path) situation. First we give the following definitions.

Feasible Comfortable Circle:

Let $P(x,y)$ denote the current position of the robotic fish, ϕ the current direction, and R_c the comfortable radius, CC a comfortable circle, $dir(x)$ an operator which represents the direction of x, bl, br the boundaries of the Left Forbidden Region and the Right Forbidden Region, the *Feasible Comfortable Circle* is defined as:

if there exists a CC passing $P(x,y)$ with direction ϕ, and exists a common tangent (ptan) between the CC and bl (or br), and $dir(ptan)$ is consistent with $dir(CC)$ and $dir(bl)$ (or $dir(br)$), then CC is called a Feasible Comfortable Circle associated with the current posture of the fish.

free path:

A path which is not obstructed by obstacles is called a *free path*.

Semi-Feasible path (shown in Fig.2(d)):

A path from the current position of the fish to the Left Attacking Point, which is composed of an arc of the Feasible Comfortable Circle, an arc of the boundary of the Forbidden Region, and the directed tangent between these two circles.

Feasible Path:

A *Feasible Path* is a *free Semi-Feasible path*.

Based on the above definitions, FP situation is described as a situation in which there exits at least one *Feasible Path* from the current position of the fish to the Left Attacking Point (see Fig. 2(d)).

(5) NFP (No Feasible Path) situation: there exits no *Feasible Path* from the current position of the fish (see Fig. 2(e)).

Associated Behavior Design. The design of the behaviors is required to lead the robotic fish to the destination posture with position the Left Attacking Point and direction the box direction, along a fine path considering the limit of the minimal turn radius of the fish.

(1) B_{SYN}: The two fish push the box coordinately since they have synchronized, while adjusting their heads toward the box direction (see Fig. 2(a)).

(2) B_{NSYN}: The NSYN situation shall be avoided with our motion planning which will be described later. Once this situation appears, the fish in position has to stop to wait for synchronizing with the other fish (see Fig. 2(b)).

(3) B_{NFAD}: In NFAD, although the fish is in the Attacking Region, it can not push the box because it has not a feasible direction. In this situation, the fish moves toward outside of the Attacking Region until a *feasible path* exits (see Fig. 2(c)).

(4) B_{FP}: The fish approaches the Left Attacking Point along the shortest *Feasible Path* (see Fig. 2(d)).

(5) B_{NFP}: The fish approaches the Left Attacking Point along the shortest *Semi-Feasible Path* while avoiding obstacle if the obstacle is near (within a specific distance r to the fish)(see Fig. 2(e)).

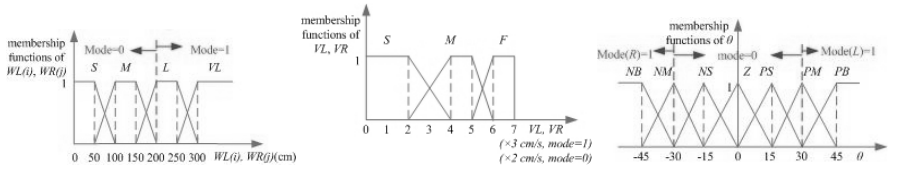
3.3 Dual-Mode Fuzzy Logic Based Motion Planning

As mentioned above, it is difficult to realize precise control of the robotic fish. To synchronize the arrival of the two robotic fish, and to move the box successfully to the goal location, we employ fuzzy logic control method to plan the motion of the fish, because rule-based fuzzy logic provides a scientific mechanism for reasoning and decision making with uncertain and imprecise information. The mode switch between FAST SWIMMING mode and SMOOTH SWIMMING mode is embedded in the fuzzy logic control to satisfy the different performance requirements.

Motion Planning in Synchronizing Procedure. Aiming to direct the two fish to reach the Attacking Points synchronously, we control the speeds of the fish when they are approaching the Attacking Points with several fuzzy rules. The inputs are $WL(i)$ and $WR(j)$, which are described as:

$$WL(i) = len(LP, P_i) + CNL_{obj}, \qquad WR(j) = len(RP, P_j) + CNR_{obj}, \tag{2}$$

in which LP (respectively RP) denotes the Left (respectively Right) Attacking Point; P_i (respectively P_j) the current position of fish i (respectively fish j, which is the other fish); NL_{obj} (respectively NR_{obj}) is 1 if there is an obstacle on the planned path from P_i (respectively P_j) to LP (respectively RP), and 0 otherwise; C is a positive constant, and $WL(i)$ (respectively $WR(j)$) represents the approximate time consuming of the fish during approaching the Left (respectively Right) Attacking Point. Let VL (respectively VR) denote the speed of the fish with role *Left* (respectively role *Right*). Firstly $WL(i)$ and $WR(j)$ are represented by the linguistic fuzzy sets $\{VL, L, M, S\}$, abbreviated from VERY LARGE, LARGE, MEDIUM, SMALL, with the membership functions (MF) shown in Fig. 4(a). VL, VR are represented by $\{F, M, S\}$, abbreviated from



(a) MF of WL, WR (b) MF of VL, VR (c) MF of The Relative Angle

Fig. 4. The Membership Functions

FAST, MEDIUM, SLOW, with MF shown in Fig. 4(b). When $WL(i)$ or $WR(i)$ is under a threshold (in our experiment, we select 200 cm), precise point to point control is required. Considering that the head of the fish oscillates significantly with big oscillating amplitude, SMOOTH SWIMMING mode (mode=0) is selected. While, when $WL(i)$ or $WR(i)$ exceeds the threshold, since the robotic fish is far from the Attacking Point, it moves to the point with FAST SWIMMING mode (mode=1), since it is unnecessary for precise point to point control. Let mode(i) denote the swimming mode of fish i, some of the fuzzy logic rules are (for space limitation, not all of them are listed here):

1) If $WL(i)$ is $VL \wedge WR(j)$ is $L \wedge WR(j) > 200$, then mode(i)=1, VL is F, mode(j)=1, VR is M;
2) If $WL(i)$ is $L \wedge WL(i) > 200 \wedge WR(j)$ is M, Then mode(i)=1, VL is F, mode (j)=0, VR is M.

The parameters of the membership functions can be derived and tuned through the experiments, and the final speeds $\hat{VL}$ and $\hat{VR}$ are computed using the Center-of-Gravity (Centroid) defuzzification method.

Motion Planning in Coordinated Box-pushing Procedure. When the two fish are both in SYN situation, they shall push the box coordinately toward the goal location with their heads. Let θ denote the relative angle of the box direction to the goal direction. The positive direction of θ is shown in Fig. 2(a). To move the box toward the goal location, we devise that the direction of the box is controlled by coordinating the speeds of the two fish, which are derived through several fuzzy rules. Here θ is the input, and VL, VR are the outputs. We represent θ by the linguistic fuzzy sets $\{PB, PM, PS, Z, NS, NM, NB\}$, abbreviated from positive big, positive medium, positive small, zero, negative small, negative medium, negative big, respectively. Let mode(L) (respectively mode(R)) denote the swimming mode of the fish on the left (respectively right) side. MF of θ are shown in Fig. 4(c). When $|\theta|$ exceeds a threshold (in our experiment, we select 30°), especially when the box is rotating away the goal direction, it is very difficult for the fish to control the box toward the goal direction with SMOOTH SWIMMING mode, although SMOOTH SWIMMING is more stable. This is because the thrust force produced by the fish with SMOOTH SWIMMING mode is not enough to change the direction of the box significantly in short time. Since the robotic fish with FAST SWIMMING mode can produce bigger thrust force, when $|\theta|$ exceeds the threshold, for example, when θ exceeds the threshold, the fish on the left side will select FAST SWIMMING mode (mode=1). Let mode(L) (respectively mode(R)) denote the

swimming mode of the fish on the left (respectively right) side. Some of the fuzzy logic rules are listed here:

1) If θ is PB, then mode(L)=1, VL is F, mode(R)=0, VR is S;

2) If θ is $PM \wedge \theta > 30$, then mode(L)=1, VL is M, mode(R)=0, VR is S.

4 Experimental Results

Typical experiment scenarios are depicted in Fig. 5. The goal location is at the top left corner of the swim tank, marked by a blue pole. If the location of the box is within a distance of 35 cm (determined by the size of the box, and the pole) to the goal location, we state that the box is moved to the goal location successfully. Fig. 5(a) shows the initial scenario, in which the two fish start out along the paths planned by CCA. The orange fish is assigned with role *Left*, while the yellow fish is assigned with role *Right*. Fig. 5(b) shows the scenario in which the two fish are trying to synchronize with different swimming speeds derived from the fuzzy logic rules. Since the yellow fish is far away the Attacking Point, it moves with FAST SWIMMING mode and a higher speed; while the orange fish swims with SMOOTH SWIMMING mode since it is near the Attacking Point, and a lower speed to wait for the yellow fish to synchronize. In Fig. 5(c), the two fish have synchronized successfully. Fig. 5(d) shows the scenario at 20.9s. Since the deflection angle of the box direction to the goal direction exceeds the threshold, the orange fish swims with FAST SWIMMING mode, while the yellow fish moves with SMOOTH SWIMMING mode, and the speeds of them are computed through the fuzzy logic rules. In Fig. 5(e), the direction of the box has been gradually changed toward the goal direction, and both fish move with SMOOTH SWIMMING mode. At 42.9s, the box has been pushed to the goal location successfully, shown in Fig. 5(f). For videos of the experiments, please visit: http://www.mech.pku.edu.cn/robot/mfbp.html.

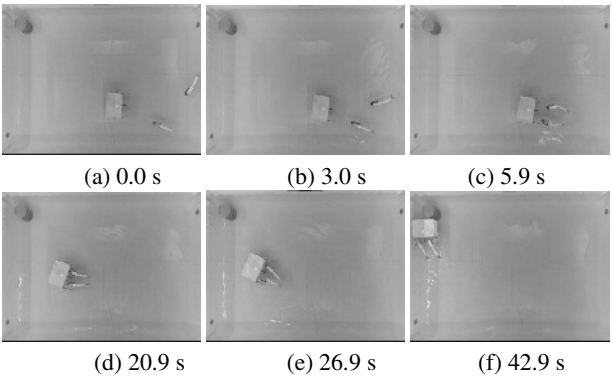


Fig. 5. Experiment Scenarios

5 Conclusion

We have presented a coordination method for multiple robotic fish box-pushing. Considering the complexity of the underwater working environment and the particularity of the movement modes of robotic fish, Comfortable Circle Approach has been proposed to simplify the path planning and action decision. In addition, dual-mode fuzzy logic method is employed to synchronize the motion of the two fish, and to control the moving direction of the box by adjusting the swimming speeds of the fish. Experimental results demonstrated that the proposed method is effective. Although the coordination method was evaluated through the box-pushing task domain, it can be utilized potentially to the object manipulation problem, or task domains which require cooperation of several smaller robots to move larger objects. In addition, our coordination method can be regarded as an improvement of Kube's work[10], in which a formalized model of cooperative transport in ants was provided and implemented on several physical robots, however, their robotic system was not very efficient. Our future research will focus on developing robotic fish with more locomotion modes, such as fast turning swimming mode, low energy swimming mode, etc. In addition, we plan to equip the fish with various sensors, so that the robotic fish is able to navigate autonomously in unknown environments and react to the environment changes flexibly. Eventually, skillfully swimming, favorably maneuverable and intelligent robotic fish will be developed, with which we can study coordination problem in more complex environments.

References

1. J. Yu, M. Tan, S. Wang, and E. Chen: Development of a biomimetic robotic fish and its control algorithm. IEEE Trans. Sys. Man and Cyber. Part B, Vol. 34. (2004) 1798–1810
2. X. Tu and D. Terzopoulos: Artificial Fishes: Physics, Locomotion, Perception, Behavior. In: ACM Computer Graphics Proceedings. Orlando, FL (1994) 43–50
3. J. Yu, and L. Wang: Parameter optimization of simplified propulsive model for biomimetic robot fish. In: Proc. IEEE Int. Conf. Robotics and Automation. Barcelona, Spain (2005) 3317–3322
4. J. Yu, S. Wang and M. Tan: Basic motion control of a free-swimming biomimetic robot fish. In: Proc. IEEE Conf. Decision and Control. Maui, Hawaii, USA (2003) 1268–1273
5. D. Barrett, M. Triantafyllou, D. K. P. Yue, M. A. Grosenbaugh, and M. J. Wolfgang: Drag reduction in fish-like locomotion. J. Fluid Mech. Vol. 392. (1999) 183C212
6. M. J. Lighthill: Note on the swimming of slender fish. J. Fluid Mech. Vol. 9. (1960) 305C317
7. Gerkey, B.P., and Mataric, M.J.: Sold!: Auction Methods for Multirobot Coordination. IEEE Trans. Robot. Autom. Vol. 18. (2002) 758–768
8. M. Sfakiotakis, D. M. Lane, and J. B. C. Davies: Review of fish swimming modes for aquatic locomotion. IEEE J. Oceanic Eng. Vol. 24. (1999) 237–252
9. D. Barrett, M. Grosenbaugh, and M. Triantafyllou: The optimal control of a flexible hull robotic undersea vehicle propelled by an oscillating foil. In Proc. IEEE AUV Symp. (1996) 1–9
10. C. R. Kube and E. Bonabeau: Cooperative transport by ants and robots. J. Robot. Auton. Syst. Vol. 30. (2000) 85–101

Effects of Spatial Growth on Gene Expression Dynamics and on Regulatory Network Reconstruction

Jan T. Kim

School of Computing Sciences,
University of East Anglia, Norwich NR4 7TJ, United Kingdom
jtk@cmp.uea.ac.uk
http://www.cmp.uea.ac.uk/people/jtk

Abstract. Morphogenesis and the spatial structure of an organism have repercussions on gene expression. These effects can influence the results of regulatory network reconstruction. An integrated, flexible and extensible computational framework for modelling gene expression dynamics within spatially growing structures is developed and used as a test system for evaluating a reconstruction algorithm. With complex morphological structures, significant effects of spatial organisation on the reconstruction process are observed. The results also reveal that stronger regulatory interactions result in more frequent cases of indirect regulation, posing a challenge for accurate network reconstruction.

1 Introduction

Regulatory gene networks are a central mechanism of organising and realising complex biological processes and structures based on genetic information. Since initial, now classical models, such as the NK model [1], there has been a steady interest in understanding regulatory networks [2,3,4,5,6,7]. High throughput "post-genomic" techniques, specifically microarrays for measuring gene expression [8,9], currently lead to renewed interest in biological networks [10,11], and various suggestions for reconstructing regulatory networks from gene expression data [12,13,14,15]. However, understanding the relation between regulatory network structure and the resulting gene expression dynamics remains a major challenge [16]. Artificial Life simulations provide a means to advance scientific understanding of gene expression dynamics in biological systems.

Complex spatial structures, which are key features of almost all biological systems. The morphology of an organism is encoded by the genome, from where it is decoded by regulatory networks. Conversely, spatial structures can have a substantial impact on gene expression dynamics. The effects of spatial growth on gene expression have to be expected to be significant for network reconstruction. In this contribution, `transsys` simulations [4] are used to explore the impact of morphogenesis and of other parameters on network reconstruction using the algorithm by Rung *et.al.* [13].

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 825–834, 2005.

2 Methods

2.1 Concept

`Transsys` networks were generated as a target of reconstruction. Various types of random network topology, as well as different characteristics of gene regulation were used for network generation. Target networks were integrated into an `L-transsys` Lindenmayer system. Development of this system was simulated for a fixed time interval, after which expression data on all genes of the target network is collected from the grown structure. For each gene in the target network, a knockout mutant was generated and gene expression values were collected. The resulting data set was used as input for regulatory network reconstruction. Reconstruction was evaluated by comparing the reconstructed network to the target network.

2.2 Modelling of Gene Expression in a Spatially Extending System

Generating Target Networks. Target networks were generated as random graphs. The number of nodes (genes) was set to 100 and the number of edges (regulatory interactions) was either set to 200 or to 500. Edges were drawn at random according to the following random network models:

NK graphs [1] are networks in which each of the N genes is regulated by K other genes, chosen at random. Thus, the incoming degree of all genes is K (either 2 or 5), while the outgoing degrees are Poisson distributed.

Random graphs are constructed by choosing each of the possible edge with equal probability. Differently from NK networks, both the incoming degree and the outgoing degree are characterised by a Poisson distribution.

Scale free graphs are characterised by a power law distribution of both incoming and outgoing degrees.

Activating and repressing edges were generated equiprobably. Networks of the same type and with the same edge density have identical topologies in this study.

For all genes, the default level of expression was set to $1.0 + \mathrm{rnd}(0.01)$, where $\mathrm{rnd}(0.01)$ denotes a random value from a uniform distribution over $[0, 0.01[$. A new random value is generated each time expression of the gene is computed. This source of variation is essential for allowing spatial gene expression patterns to form. The default expression level is subject to modification by activation or repression.

Activation and repression are described by two parameters, the maximum amount of regulation $a_{\max}$, and a_{spec}, which is the concentration of the regulator at which activation amounts to $a_{\max}/2$. These parameters were set to the same values for all edges in a network. a_{spec} was set to 0.1 in all simulations whereas $a_{\max}$ was chosen from $\{0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0\}$. The parameters for repressing edges, $r_{\max}$ and r_{spec}, were always set to the corresponding activation parameters.

The decay rate of all factors in the target networks was set to the same value. Simulations were run with decay rates of 0.8 and 0.2. For the diffusibility parameter, values of 0.1 and 0.3 were tested.

Fig. 1. Final growth stages for the single shoot structure (left) and the *Arabidopsis thaliana* model (right). Red and blue spheres represent represent meristems, i.e. growth centres from which new morphological elements are generated.

Embedding Networks into `L-transsys` Systems. The process of morphogenesis within which gene expression dynamics controlled by the target network takes place is externally specified in the scenarios studied here. Three morphological structures, depicted in Fig. 1, are used. The cell structure consists of just one symbol, the cell, and no L-system rules. Therefore, no spatially extended structure develops. This serves as a control. The single shoot structure starts out with one meristem, which produces a phytomer consisting of a leaf and an internode (a stem piece) every 20 time steps. The *Arabidopsis* structure is a rather coarse-grained `L-transsys` model of *Arabidopsis* growth, proceeding through stages of rosette leaf growth with decussate and spiral phyllotaxis, bolting, and flower formation.

All three structures are specified by a growth controlling `transsys` program and an `L-transsys` specification. The factors and genes of the target network are inserted into the `transsys` program. The target network does not have any effect on morphogenesis, but growth of the plant structure does have effects on gene expression dynamics. This ensures that all measurements of knockout mutants are based on identical morphological structures.

This approach simulates reconstruction of a target network that does not organise morphogenesis, but may be informed by it. It was chosen here to enable attribution of differences to individual morphological structures, rather than to collections of mutant structures with complex and unfavourable statistical properties. For example, if morphogenesis was controlled by the target network, there may not be any growth in a significant fraction of knockout mutants. Such non-growing mutants would be equivalent to the single cell structure, and consequently, differences between the single cell and the more complex morphological structures would be blurred.

Simulation of Gene Expression Measurement. 504 cases, resulting from combination of 3 structures, 3 random graph types, 2 edge densities, 7 regulatory strength settings, 2 decay settings and 2 diffusibility settings, were assayed. Each structure was grown for 250 time steps, starting out with a single symbol. After

the final time step, the expression level of all factors of the target network, averaged over all symbols, was measured. The resulting vector of gene expression levels corresponds to one microarray experiment in molecular biology.

2.3 Reconstruction of Regulatory Networks

The method for regulatory network reconstruction introduced by Rung *et.al.* [13] uses a set of mutants, called knockout mutants, each of which has one gene disrupted such that it encodes a non-functional gene product. For each knockout mutant, and for the wild type as a control, expression of all genes is measured. The results are assembled into an expression data matrix in which the elements r_{ij} denote the logarithm of the ratio of the expression level of gene i in the mutant with gene j disabled to the expression level of gene i in the wild type. Subsequently, normalised values $\tilde{r}_{ij} = r_{ij}/\hat{\sigma}_{ij}$ are computed, where $\hat{\sigma}_{ij}$ are estimated standard deviations. As measurement is undistorted in the simulations, this step was effectively omitted by setting all $\hat{\sigma}_{ij} = 1$.

The regulatory network graph is then reconstructed by starting with the genes as isolated nodes and placing an edge from gene j to gene i if $|\tilde{r}_{ij}| \geq \gamma$, where γ is a user-supplied threshold. For negative values of $\tilde{r}_{ij}$ (gene i expressed at lower level in absence of gene j), an activating effect is predicted.

2.4 Analysis of Reconstruction

The threshold γ controls the sensitivity and the specificity of the reconstruction algorithm by Rung *et.al.*, low values providing a high sensitivity but low specificity while high threshold settings give good specificity at the expense of a low sensitivity. Since choice of the threshold value is arbitrary in the sense that it is not systematically deduced from the expression data, ROC (Receiver-Operator-Characteristic) curves were used to assess the performance of reconstruction. A ROC curve is computed by reconstructing networks with different threshold settings, ranging from 0 to $\max_{i,j} |\tilde{r}_{ij}|$. For each threshold, sensitivity and specificity are determined. Connecting these points yields the ROC curve. The area under the curve indicates the potential of of the reconstruction procedure. A value of 1 means that perfect reconstruction is possible while a value of 0.5 indicates no potential. By integrating over all possible threshold values, this approach allows assessing the reconstructive potential independently of γ.

To further investigate the reconstruction process, the length of the shortest connecting path, denoted by p_{ij}, was computed for all pairs (i, j) of genes in the target network. For this purpose, no difference between activating and repressing regulatory connections was made, both types were treated as directed edges with length 1. For perfect reconstruction to be possible, there has to exist a γ such that $\forall(i, j) : p_{ij} = 1 \Leftrightarrow \gamma \leq |\tilde{r}_{ij}|$. A scatter plot of all pairs $(p_{ij}, \tilde{r}_{ij})$ reveals whether this condition is satisfied, and provides further insight into the effects which the different variants of network structures and the parameters controlling gene expression dynamics have on the performance of network reconstruction.

2.5 Software

Generation of knockout mutants and collection of gene expression measurements was implemented in `Python`,[1] based on the `transsys` framework [4]. `R` [17] was used for programming data analysis and visualisation. The code underlying the results presented here will be made available on the `transsys` website,[2] which also provides further information on technical aspects of `transsys`.

3 Results and Discussion

3.1 Effects of Spatial Structure

Fig. 2 shows ROC curves for a random graph network, expressed in the single cell and in the *Arabidopsis* structure. A clear difference between reconstruction is observed. In the single cell case, almost perfect specificity is possible up to a sensitivity of 0.95, while a significant decline of specificity is seen with the *Arabidopsis* structure even at sensitivity levels below 0.5.

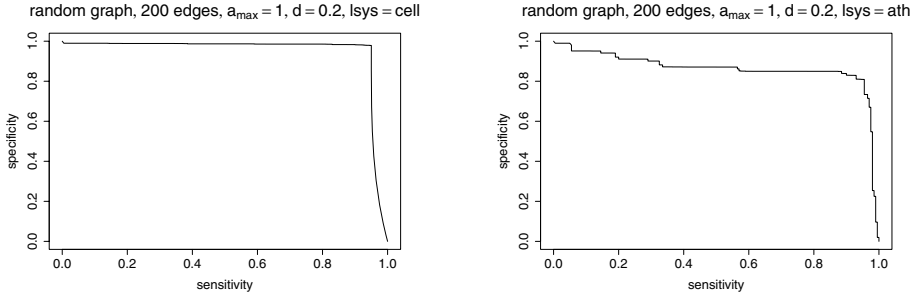


Fig. 2. ROC curves showing the reconstructive potential for a random graph with $N = 100$ genes and 200 edges, regulatory strength $a_{\max} = 1$ and a decay rate of 0.2. Left: results with a single cell, right: results with the *Arabidopsis* structure.

The scatter plot of p_{ij} vs. $\tilde{r}_{ij}$, shown in Fig. 3, reveals that this decline in specificity is due to a substantial increase of variance in the $\tilde{r}_{ij}$ values. There are gene pairs i, j separated by up to 9 network links (p_{ij} up to 9) and $\tilde{r}_{ij} > 2$ observed in the *Arabidopsis* structure, while in the single cell, $-0.297 \leq \tilde{r}_{ij} \leq 0.297$ for all gene pairs with $p_{ij} > 4$. There is a clear trend that effects of a gene knockout on the expression level are more pronounced if the disabled gene is close within the regulatory network, but effects on genes that are distant in the network are possible as well.

While there are substantial differences between reconstruction based on the single cell and the *Arabidopsis* structure, the results obtained with the shoot and the cell structures were not significantly different, as summarised in

[1] `http://www.python.org/`
[2] `http://www.cmp.uea.ac.uk/~jtk/transsys/`

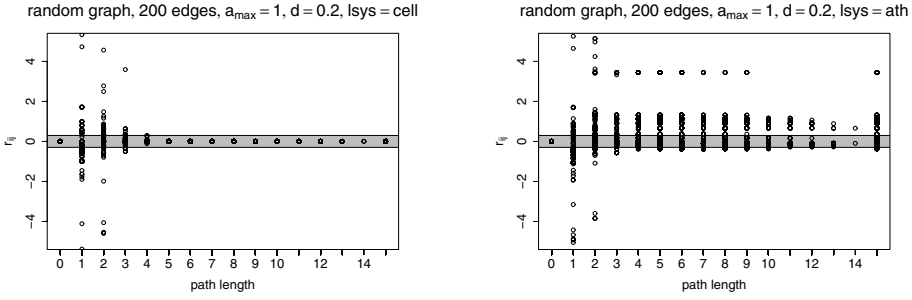


Fig. 3. Scatter plots of path length p_{ij} vs. normalised log ratios $\tilde{r}_{ij}$ for the same target network as in Fig. 2. Left: results with a single cell, right: results with the *Arabidopsis* structure. The gray area shows the range $[-\gamma, \gamma]$ of $\tilde{r}_{ij}$ where no edge is predicted, for $\gamma = 0.297$, the optimal γ value for the *Arabidopsis* structure.

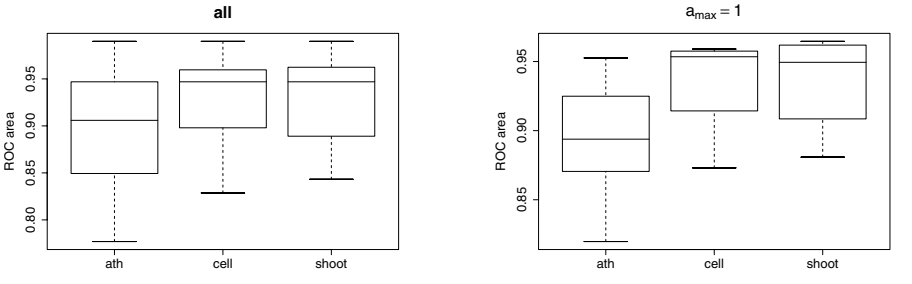


Fig. 4. Box-plots summarising reconstruction performance for the cell, shoot and *Arabidopsis* structures. Boxes show interquartile range of the area under the ROC curve, the horizontal line within each box shows the median value. The differences between the structures are more pronounced with stronger regulatory interactions, as exemplified by $a_{\max} = 1$ (right plot).

Fig. 4. The box-plots show a significantly lower reconstructive potential with the *Arabidopsis* structure, as exemplified by the case discussed above, is generally observed with strong regulatory effects, provided by $a_{\max} = 1$.

Reconstruction performance was generally similar with the single cell and the single shoot structure. This observation indicates that more complex spatial structures have more pronounced effects on regulatory dynamics and network reconstruction.

3.2 Effects of Regulation Strength

Regulation strength has a major impact on reconstruction. The scatter plots shown in Fig. 5 show that with weak regulation with $a_{\max}$ up to 0.4, there are no significant indirect regulatory effects: For all gene pairs with $p_{ij} > 1$, the corresponding value of $\tilde{r}_{ij}$ does not substantially deviate from 0. Consequently, very accurate reconstruction can be achieved.

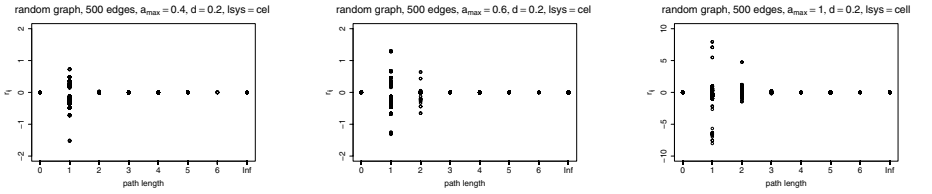


Fig. 5. Scatter plots of path length p_{ij} vs. normalised log ratios $\tilde{r}_{ij}$ for an random graph network with $N = 100$ genes and 500 edges. Left: $a_{\max} = 0.4$, middle: $a_{\max} = 0.6$, right: $a_{\max} = 1$. Notice the different scale of the $\tilde{r}_{ij}$ axis in the right plot.

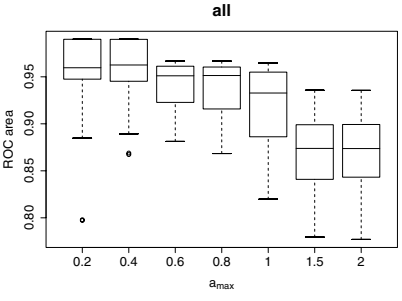


Fig. 6. Box-plot summarising reconstruction performance as a function of regulatory strength $a_{\max}$

With $a_{\max} = 0.6$, significant indirect regulatory effects occur, making perfect reconstruction impossible, as the algorithm by Rung *et.al.* cannot distinguish direct from indirect effects. With $a_{\max} = 1$, regulatory effects increase by an order of magnitude and more overlap between direct and indirect effects that result in $\gamma \leq \tilde{r}_{ij}$ results. Overall, this results in a decline of reconstruction potential as $a_{\max}$, and hence the extent of indirect regulation, increases, as summarised in Fig. 6.

3.3 Effects of Network Structure

The effects of network structure on reconstruction are summarised by the exploits in Fig. 7. This analysis was restricted to the samples with stronger regulation ($a_{\max} \geq 1$) because this reveals effects that are obscured by the large number of cases of near perfect reconstruction if the entire data set is included.

For the graphs with 200 edges, the median reconstruction potential achieved for the three networks exhibits different levels of variance, but the median reconstruction potential is very similar. In contrast to this, the networks with 500 edges result in different reconstruction potentials; the random graph can be reconstructed significantly better than the other types.

Considering that only one representative of each type and density has been evaluated, the results presented here are not sufficient for a deeper analysis of the effects caused by network structure. It is, however, interesting to note that

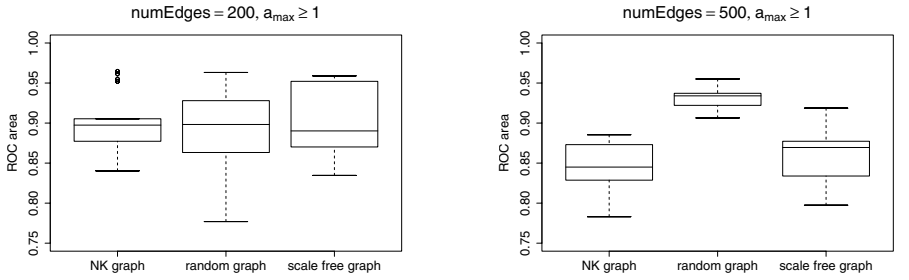


Fig. 7. Box-plots summarising reconstruction performance for sparse (200 edges, left plot) and the dense (500 edges, right) variants different types of networks (NK graph, random graph and scale free graph), for networks with $a_{\max} \geq 1$

effects of network structure on reconstruction performance depend quite strongly on parameters of gene expression such as strength of regulation.

4 Conclusion and Outlook

Artificial Life simulations provide a basis for evaluating methods to reconstruct regulatory networks based on gene expression measurements. Here the `transsys` framework was used to investigate the reconstruction method by Rung *et.al.*

This algorithm assumes that significant changes in expression levels resulting from a gene knockout indicate a direct target gene. The results presented here show that indirect regulation is more frequent in systems with stronger regulatory effects. It would therefore be important to develop criteria for estimating the extent of indirect regulation, and to further develop methods to identify cases of indirect regulation to refine reconstruction by improving specificity.

The results presented here indicate that embedding gene expression within a complex, growing spatial structure results in the formation of patterns that are different from those observed with the same dynamical system within a spatially unstructured environment. This is, in fact, a classical topic in Artificial Life [18,19]. In the interest of focusing on the effects of spatial growth on gene expression dynamic and on network reconstruction, the converse effects of gene expression on morphogenesis have been excluded in this study. This model approximates the case of subnetworks that realise functions other than morphogenesis, but it does not adequately capture networks that organise morphogenesis. This important case will be addressed by studying networks evolved to control morphogenesis. An evolutionary model, based on `LindEvol` [3], is currently being developed for this purpose.

The framework presented here provides points of departure for various further studies. More detailed studies of the impact of decay and diffusion on expression dynamics and network reconstruction are currently underway, and the spectrum of network topologies and dynamical parameters will be further extended. Studies of noise effects will use simulation of measurement errors, as described in [20], and employ the standard deviation of gene expression within the spatial

structures as a measure for biological noise. This will allow to include the normalisation step used by Rung *et.al.* in the evaluation, and make the framework more useful for testing other reconstruction methods.

References

1. Kauffman, S.A.: Developmental logic and its evolution. BioEssays **6** (1987) 82–87
2. Reil, T.: Dynamics of gene expression in an artificial genome – implications for biological and artificial ontogeny. In Floreano, D., Nicoud, J.D., Mondada, F., eds.: Advances in Artificial Life. Lecture Notes in Artificial Intelligence, Berlin Heidelberg, Springer-Verlag (1999) 457–466
3. Kim, J.T.: Lindevol: Artificial models for natural plant evolution. Künstliche Intelligenz (2000) 26–32
4. Kim, J.T.: `transsys`: A generic formalism for modelling regulatory networks in morphogenesis. In Kelemen, J., Sosík, P., eds.: Advances in Artificial Life (Proceedings of the 6th European Conference on Artificial Life). Volume 2159 of Lecture Notes in Artificial Intelligence., Berlin Heidelberg, Springer Verlag (2001) 242–251
5. Banzhaf, W.: On the dynamics of an artificial regulatory network. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J.T., Ziegler, J., eds.: Advances in Artificial Life (ECAL 2003). Volume 2801 of Lecture Notes in Artificial Intelligence., Berlin Heidelberg, Springer Verlag (2003) 217–227
6. Bongard, J.: Evolving modular genetic regulatory networks. In Fogel, D.B., El-Sharkawi, M.A., Yao, X., Greenwood, G., Iba, H., Marrow, P., Shackleton, M., eds.: Proceedings of the IEEE 2002 Congress on Evolutionary Computation (CEC2002), Piscataway, NJ, IEEE Press (2002) 1872–1877
7. Bornholdt, S., Rohlf, T.: Topological evolution of dynamical networks: Global criticality from local dynamics. Physical Review Letters **84** (2000) 6114–6117
8. Lee, T.I., Rinaldi, N.J., Robert, F., Odom, D.T., Bar-Joseph, Z., Gerber, G.K., Hannett, N.M., Harbison, C.T., Thompson, C.M., Simon, I., Zeitlinger, J., Jennings, E.G., Murray, H.L., Gordon, D.B., Ren, B., Wyrick, J.J., Tagne, J.B., Volkert, T.L., Fraenkel, E., Gifford, D.K., Young, R.A.: Transcriptional regulatory networks in *Saccharomyces cerevisiae*. Science **298** (2002) 799–804
9. Yu, H., Luscombe, N.M., Quian, J., Gerstein, M.: Genomic analysis of gene expression relationships in transcriptional regulatory networks. Trends in Genetics **19** (2003) 422–427
10. Bray, D.: Molecular networks: The top-down view. Science **301** (2003) 1864–1865
11. Barabási, A.L., Oltvai, Z.N.: Network biology: Understanding the cell's functional organization. Nature Reviews Genetics **5** (2004) 101–113
12. Akutsu, T., Miyano, S., Kuhara, S.: Inferring qualitative relations in genetic networks and metabolic pathways. Bioinformatics **16** (2000) 727–734
13. Rung, J., Schlitt, T., Brazma, A., Freivalds, K., Vilo, J.: Building and analysing genome-wide gene disruption networks. Bioinformatics **18** (2002) S202–S210
14. Repsilber, D., Liljenström, H., Andersson, S.G.: Reverse engineering of regulatory networks: Simulation studies on a genetic algorithm approach for ranking hypotheses. BioSystems **66** (2002) 31–41
15. Bongard, J., Lipson, H.: Automating genetic network inference with minimal physical experimentation using coevolution. In ?, ed.: Proceedings of the 2004 Genetic and Evolutionary Computation Conference (GECCO), Berlin Heidelberg, Springer Verlag (2004) 333–345

16. Variano, E.A., McCoy, J.H., Lipson, H.: Networks, dynamics and modularity. Physical Review Letters **92** (2004) 188701
17. R Development Core Team: R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. (2004) ISBN 3-900051-07-0.
18. Boerlijst, M., Hogeweg, P.: Self-structuring and selection: Spiral waves as a substrate for prebiotic evolution. In Langton, C.G., Taylor, C., Farmer, J.D., Rasmussen, S., eds.: Artificial Life II. Volume X of Santa Fe Institute Studies in the Sciences of Complexity, Proceedings., Redwood City, CA, Addison-Wesley (1992) 255–276
19. Kumar, S., Bentley, P.J., eds.: On Growth, Form and Computers. Elsevier Academic Press, Amsterdam (2003)
20. Repsilber, D., Kim, J.T.: Developing and testing methods for microarray data analysis using an artificial life framework. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J.T., Ziegler, J., eds.: Advances in Artificial Life (ECAL 2003). Volume 2801 of Lecture Notes in Artificial Intelligence., Berlin Heidelberg, Springer Verlag (2003) 686–695

Evolution of Song Communication in a 2D Space

Kazutoshi Sasahara[1] and Takashi Ikegami[2]

[1] Laboratory for Biolinguistics, Brain Science Institute, RIKEN,
2-1 Hirosawa, Wako, Saitama 351-0198, Japan
[2] Department of General Systems Studies, Graduate School of Arts and Sciences,
University of Tokyo, 3-8-1, Komaba, Meguro-ku, Tokyo 153-8902, Japan
sasahara@brain.riken.jp

Abstract. Song communication of artificial birds is simulated in a 2D space, in which male and female birds communicate and then leave their offspring based on their communication performance. The communication is modeled as interaction between different types of finite-state automata, one for song production by males, the other is for song evaluation by females. In addition, an abstract space is introduced for studying how spatial structure affects the evolution of song communication system. We find a correlation between global spatiotemporal patterns and local communications between artificial birds. In particular, we report a habit segregation phenomenon of our simple ecosystem.

1 Introduction

Recent studies of a particular songbird have shown that males have complex rules for singing (henceforth, *song grammar*), which are represented by finite-state automata (FAs). In addition, males that can sing complex songs tend to be preferred by females [9, 13, 17] These findings give us an insight that the complexity of song rules can be driven not only by *natural selection* but also by *sexual selection* [13].

So far, sexual selection has been studied in a variety of ways focusing on different aspects [2,8,11]. By modeling song communication, we have been studying the co-evolution of males and females modeled as two types of FA, interacting (communicating), replicating (producing offspring) in a simple ecosystem. With a homogeneous space (e.g. bird-cage), in which every male has an equal chance to communicate with every female, it has been clearly demonstrated that the complexity of song grammars could evolve as a result of the diversity of female preferences [14, 15]. However, due to the lack of spatial structure, open-ended evolution is missing in our model; for example, the males with similar complexity of song grammars occupied the ecosystem in the later stages of the evolution.

The importance of space for open-ended evolution has been indicated in some simulation studies. [6, 5, 12]. One of the main reasons is that a model with a spatial structure allows unbounded evolution of strategies. For example, Lindgren (1994) demonstrated the cooperation and community structure in an artificial ecosystem by considering the iterated Prisoner's Dilemma game in a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 835–844, 2005.

2D lattice space [6]. Some field research of song birds has also been focused on spatial structure investigating geographical variation in songs [16]. To study diverse evolution of song grammars, we extend the previous model by introducing a 2D space. We study the correlation between the spatiotemporal patterns of birds and their evolving communication features. Finally we discuss essential differences in evolution between a homogeneous and a 2D space.

2 Modeling

2.1 Song Communication

We introduce the following song communication along with the previous homogeneous space model, which was inspired by experimental observations [3, 15]. Real female finches do not sing during communication with males. Instead they may synchronize their body motion with the males' song. In our artificial model, a male sings according to his song grammar and a female inserts a special symbol when the male pauses. This behavior of the females we call *interjection*.

In sexual selection, *novelty* may play a key role because courtship songs seem to be sexual displays for tempting females [7,4,8]. So we assume a song to which a female can interject perfectly is boring for her, i.e. has a lack of novelty. In other words, a female must make at least one mistake in interjection in our model (henceforth, this is called the *novelty condition*). Unless the novelty condition is satisfied in courtship songs, the males involved are not eligible candidates for mating.

2.2 Artificial Birds

To model the song communication described above, two different types of automaton (FA) are employed. A male has a song grammar modeled as a sequential machine: $G = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$, where Q is a finite set of states, q_0 is an initial state, Σ is a finite set of input symbols, Δ is a finite set of output symbols, δ is a state transition function, $Q \times \Sigma \to Q$, λ is an output function and $Q \times \Sigma \to \Delta$ [1]. In this model, $\Delta = \{blank, A, B, ..., J\}$, where each letter denotes a song chunk and '*blank*' denotes a silent interval between chunks. A male sings (outputs chunks) in accordance with his song grammar G. A collection of combinatorial chunks between blanks represents a phrase and the whole output sequence expresses a courtship song.

To estimate the structural complexity of a song grammar, the linearity (LI) of a FA is defined as $N^{male}_{node}/N^{male}_{arrow}$, where N^{male}_{node} denotes the number of nodes and N^{male}_{arrow} denotes the number of arrows leaving from a node. If $N^{male}_{node} = N$, this value ranges between $1/N \leq LI \leq 1$ as N_{arrow} varies from N^2 to N. Thus more complex song grammars have lower values of LI.

On the other hand, a female has a preference P expressed by a FA that determines the timing of interjection: $P = (Q, \Sigma, \delta, q_0, F)$. Here Q, Σ, δ, and q_0 are the same as above, and F is a set of accepting states, which is a subset of Q. A female changes her internal state by listening to a song (receiving a

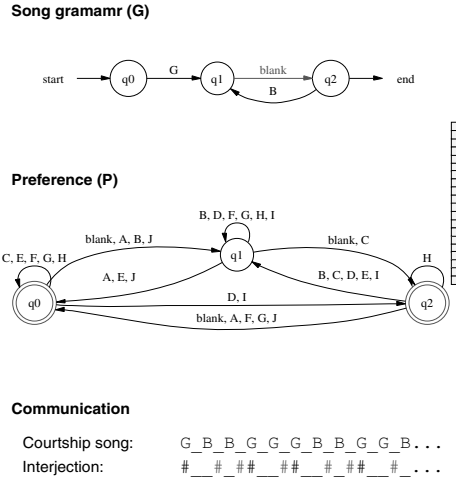


Fig. 1. Example of song grammar, mating preference and communication. The top FA represents a male's song grammar (G) and the middle FA represents a female's preference (P) for courtship songs. A communication of these birds is illustrated in the bottom. '#' denotes a interjection and '_' denotes a silent interval.

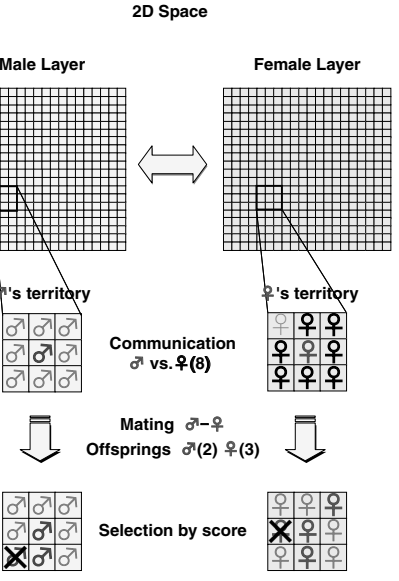


Fig. 2. Schematic representation of the 2D simulation. The space consists of two lattice layers. One bird may dominate each cell.

song sequence as an input) and interjects when her internal state becomes an accepting state. When a female hears her favorite phrases (favorite arrangements of chunks) and interjects to the breaks, *blank*, of a song while listening, it is deemed as a successful interjection. An example of FA of a male and a female and their song communication are illustrated in Fig.1.

2.3 Spatial Structure

We introduce a space consisting of a square lattice (i.e. Moore neighborhood) with rigid boundary. The entire space has a double-layered structure; a male and a female layer are constructed on the lattice (see Fig.2). Then males and females live in their respective layers. Only one bird may dominate each cell of each layer. In each layer, the Moore neighborhood of a bird is considered his or her territory.

2.4 Courtship and Mating

A male selects a female randomly from his territory and sings to her for song length L_{song} in accordance with his song grammar G. In total, a male sings for up to L_{song}^{total}, where L_{song}^{total} denotes the total length he can sing. For example,

if a male has $L_{song} = 10$ and $L_{song}^{total} = 80$, he sings to eight females within his territory in total. In a single time step, half of all males randomly selected behave as mentioned above.

In a communication with a male, a female hears his song by interjecting with her preference P. In proportion to the successful interjection (N_{interj}^{succ}), the communication score is assigned as follows:

$$S = \frac{1}{N_{interj}^{thresh}} min(N_{interj}^{succ}, N_{interj}^{th}) + \frac{N_{interj}^{succ}}{N_{interj}^{all}} + \frac{N_{chunk}}{L_{song}} \quad (1)$$

where, $0 \leq S < 3$. The first term denotes the evaluation of the number of successful interjections; when $N_{interj}^{succ} \geq N_{interj}^{thresh}$, this term becomes one (i.e. the female's evaluation is saturated). The second term denotes the success rate of interjection. The third term denotes the fraction of non-empty chunks in a song. In total, the communication score considers the evaluation of both quantity and quality of interjection, and the richness of chunks. A female evaluates all songs that she hears. According to the communication scores, each female selects the male with which she has the highest communication score as a mating partner.

In this model, polygamy is adopted as a mating style. Thus a male favored by many females can leave many offspring; the sons inherit song grammars similar to their father's.

2.5 Evolution

The number of offspring is calculated as $C_{offs} \cdot S$, which is proportional to the communication score. A female produces offspring in either her or her mating partner's territory. For example, if a female has two sons and three daughters, she randomly leaves the daughters in her territory and the sons in that of her partner (see Fig.2).

Then their offspring's genders are randomly assigned and they are added into the ecosystem as new child birds. Since child birds learn songs from their fathers or may have similar song preferences to their mothers as a result of their upbringing, their characters become similar to those of their parents. In our model, therefore, the child birds inherit FAs similar to their parents, changed according to one of the following genetic mutation operations (the frequency of each mutation is the same):

(a) **Arrow mutation:** Select an arrow of the FA at random and change the transition with the number of nodes remaining fixed. The probability of adding a new transition is $1/N_{chunk}^{total}$, where N_{chunk}^{total} is fixed at 11.
(b) **Node mutation:** Change the number of nodes (± 1) and then add or remove arrows as required. The probability of adding a new transition is same as above.
(c) **Random mutation:** A new FA is made at random.

These (a)-(c) express the possible inaccuracy in child birds inheritance of their parents' characteristics, song grammars G and preferences P. In particular, the

accuracy of inheritance is highest in (a). On the other hand, (c) represents complete failure to inherit any characteristics from the parents.

Moreover, the following mutation is performed for the male child birds:

(d) **Song length mutation:** Change L_{song} (± 5), and change L_{song}^{total} (± 2)

After both production and mutation of all offspring, selection is performed to all birds. In parent birds, it depends on the highest scores they got; in child birds, it depends on the score their parents got. Then, in every cell of the male layer, only the male with the highest score can survive. The same holds for females.

3 Simulation Results

We show some typical results of our 2D simulations. At the initial state, every male and female had a FA constructed randomly with $N_{node} = 2$. A 100×100 lattice was adopted, and 10000 individuals were set in the male and female layer, respectively. The other parameters are listed in table 1. To aid comparison, these are almost the same as those used previously [15].

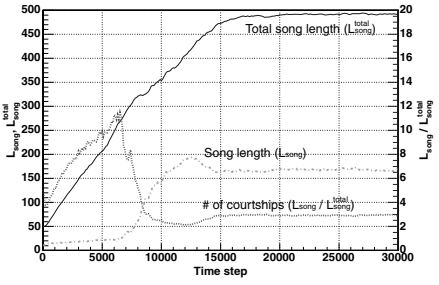
Table 1. Parameter setting

Initial males	$N^{male} = 10000$, $N_{node}^{male} = 2$, $L_{song} = 10$, $L_{song}^{total} = 50$
Initial females	$N^{females} = 10000$, $N_{node}^{female} = 2$
Communication	$N_{chunk}^{total} = 11$, $N_{interj}^{th} = 100$ (c.f.(1)), $max(L_{song}^{total}) = 500$; half of all males communicate in one time step
Mating	$C_{offs} = 1.5$; polygamy
Spatial structure	a 100×100 lattice (rigid boundary) $\times 2$

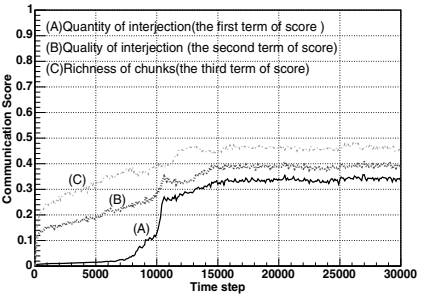
3.1 Global Dynamics

A global dynamics of the 2D model is shown in Fig.3: (a) communication features, (b) communication score, (c) the number of nodes of males' and females' FAs, (d) song complexity. In Fig.3(a), the average total song length (L_{song}^{total}) linearly increased toward the maximum length (500) and then remained saturated. The average song length (L_{song}) was suppressed under 40 during the early stages of the evolution. Around $t = 7800$, then, L_{song} rapidly increased to about 200 for the next 5000 time steps and settled down to about 160. Note that the ratio of L_{song} to L_{song}^{total} represents the number of male courtships each time step. The change of $L_{song}^{total}/L_{song}$ indicates that most of the males had been singing short songs to many females in the early stages of the evolution; whilst after $t = 7800$ they had been singing longer songs to fewer females. This is considered as a change in courtship strategies depending on song length.

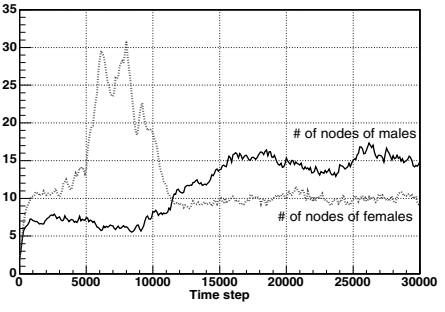
The communication score also increased after $t = 7800$ in Fig.3(b). In particular, a component of the score, 'the quantity of interjection' (dotted line), abruptly increased after the emergence of males with longer songs. Such courtship strategies were observed in our previous simulations of a homogeneous (bird-cage)



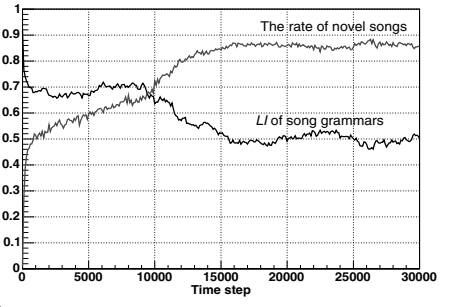
(a) Song length



(b) Communication score



(c) The number of nodes in males' and females' FAs



(d) Song complexity

Fig. 3. Global dynamics of the 2D space model: (a) Males which could sing longer songs emerged at about $t = 7800$. (b) The quantity of successful interjection increased rapidly after the emergence of the males with longer songs. (c) Complex females with a bunch of nodes were observed from $t = 5000$ to 10000. (d) The complexity and novelty of songs increased gradually over time.

space, although much more time was needed for the appearance of males with longer songs [15].

However, the evolved females look different from ones obtained in the previous model. If we look at the average number of nodes in males and females (N_{node}^{male} and N_{node}^{female}) in Fig.3(c), most of the females developed FAs with about $N_{node}^{female} = 10$ before $t = 5000$. Then, the females with a large number of nodes had been dominating the space from $t = 5000$ to 10000 (also see Fig.4). Note that when females' FAs have many internal states (i.e. many nodes), their preferences can be very complex. Such complex females were not observed in the previous model. Unexpectedly, after $t = 10000$, the complex females disappeared.

In Fig.3(d), we see that the structural complexity of song grammars was increasing (i.e. LI was decreasing). This trend was accompanied by the emergence of males with longer songs. The number of novel songs also increased according to the alternation of courtship strategies. So far, we have seen that the change of courtship strategies and song complexity depends on song length.

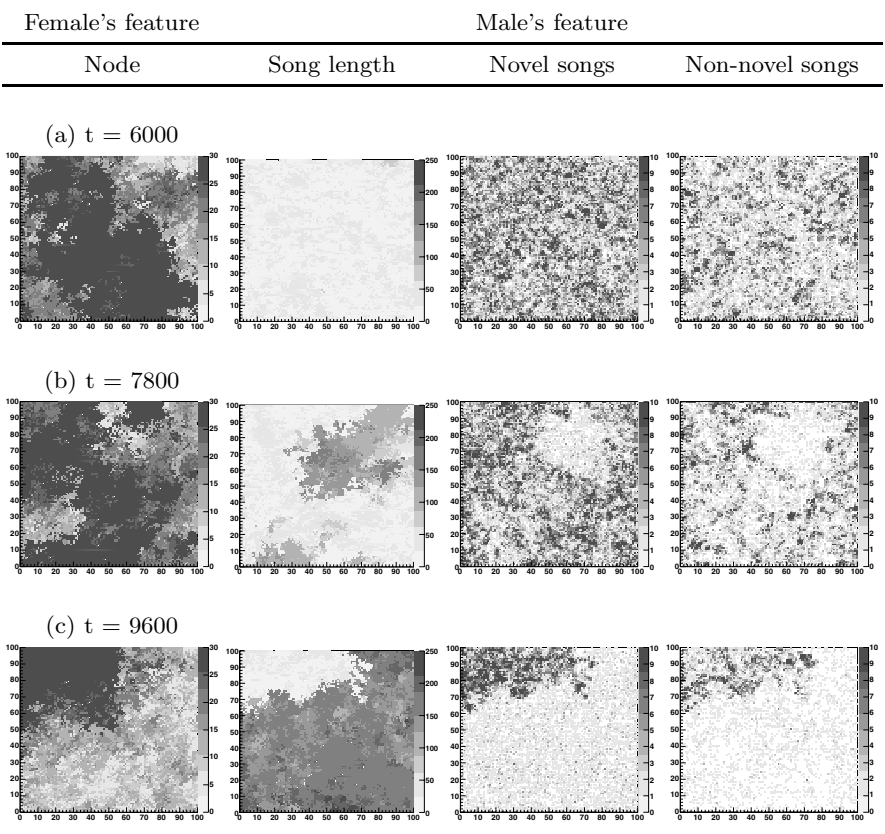


Fig. 4. Communication features as spatiotemporal patterns. The vertical and horizontal axis is the position of birds. (a) There is no correlation in communication features of males and females. (b) The male and female layer become correlated after $t = 7800$. (c) Habitat segregation of birds is clearly observed in between the number of nodes of females and the song length of males.

3.2 Spatiotemporal Patterns and Communications

Fig.4 shows snapshots of the spatiotemporal patterns of communication features: the number of nodes in females' FAs, song length and the number of both novel and non-novel songs. Males which could sing longer songs did not exist in the male layer at $t = 6000$. Meanwhile, complex females with many nodes dominated the female layer. At this time, there was no correlation between the spatiotemporal patters of the song length of males and the number of nodes in females' FAs.

As the males with longer songs dominated the space, the spatiotemporal patterns of communication features mentioned above became correlated. For example, similar patterns are observed in both N_{node}^{female} and L_{song} in Fig.4(b); that is, the males with shorter songs and the females with many nodes co-evolved,

dominating the same part of the space. Such a correlation became prominent at $t = 9600$, at which the '*habitat segregation phenomenon*' was observed. Fig.4(c) clearly shows the coexistence of specific males and females, in which the males with shorter songs and complex females with many internal states (i.e. with a high number of nodes) aggregated in the upper-left corner of the space while the males with longer songs and simple females with a moderate number of nodes ($N_{node}^{female} \sim 10$) occupied the remaining area.

This habitat segregation was largely concerned with the emergence of longer songs because it happened during the transition period from shorter to longer songs. After $t = 10000$, the habitat segregation phenomenon began to be disrupted, and consequently the males with longer songs and simple females began to dominate the space. In the later stages of the evolution, such a phenomenon was not observed.

3.3 Complexity of Mating Preferences

Fig.5 shows histograms of the total communication score of simple females ($N_{node}^{female} < 10$) and complex females ($N_{node}^{female} > 30$) at $t = 9600$. It is interesting to note that during the habitat segregation, the simple and complex females obtained comparable scores; $S = 1.3 \pm 1.2$ and 1.0 ± 1.2, respectively. However, they differ in the distribution of scores; the histogram of the complex females has three peaks, while the simple females' histogram has one peak. Each peak reflects a different strategy of successful interjection depending on males' song length.

In Fig.4(c), novel and non-novel songs were both aggregated in the upper-right area where females could distinguish them by having a large number of internal states (nodes) in their FAs. The reason is that such females are sensitive to the order of inputs, so that they can predict song inputs better than those with a few nodes, if songs are short enough. Those females could therefore leave their offspring successfully.

On the other hand, we see from Fig.6 that the simple females ($N_{node}^{female} < 10$) tended to get higher scores (more than six) for longer songs at $t = 9600$. This shows that simple females having a moderate number of nodes (about 10) were advantaged in successful interjection to longer songs. The habitat segregation phenomenon broke down as the offspring of the simple females dominate the space.

4 Discussion

By introducing a 2D space structure, we demonstrated the evolution of song communication of artificial birds. We focused on the global spatiotemporal patterns and local communication features in artificial birds. The global behaviors of the homogeneous model and the 2D spatial model had a number of common features. For example, as with the previous model, males with longer songs emerged thereby changing courtship strategies and song complexity in the evolution of males [15]. This shows that the homogeneous model is an adequate mean field approximation of the 2D spatial model.

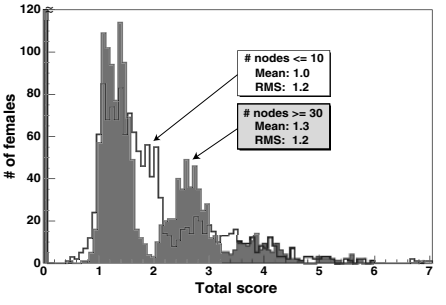


Fig. 5. Comparison of total score between two female populations. The average total scores are approximately same regardless of N_{node}^{female} at $t = 9600$. However, the distribution is different.

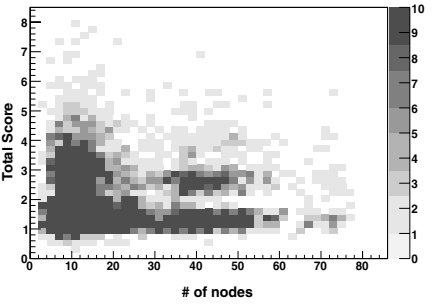


Fig. 6. Relationship between the total communication score and the complexity of females. Simple females tend to acquire much greater total scores than complex ones at $t = 9600$.

On the other hand, there were also significant differences from the previous model; evolution in the 2D space was more diverse than in the homogeneous model. For example, we observed the transition from complex females ($N_{node}^{female} > 30$) to simple ones ($N_{node}^{female} < 10$). Such complex females with many nodes were not observed in previous simulations. Furthermore, the habitat segregation lasted for about 2000 time steps in the evolution. During this period, males with shorter songs and complex females had been dominating the upper-left region of space while the males with longer songs and simple females dominated the rest of the space.

We did not assume any inhomogeneity in the 2D space. The co-existence and co-prosperity of the specific males and females did not result from geographic variation (e.g. obstacles or other discontinuities). Instead, the habitat segregation was self-organized and maintained dynamically by song communication itself. Our simulations illustrate that a spatial structure can enhance the diversity of song communication system.

Acknowledgements. This work was partially supported by Grant-in aid (No.09640454) from the Ministry of Education, Science, Sports and Culture, the 21st Century COE (Center of Excellence) program (Research Center for Integrated Science) of the Ministry of Education, Culture, Sports, Science, and Technology Japan and the Takumori Foundation. KS would like to thank Prof. Okanoya for his helpful comments.

References

1. Hopcroft, J. E. and Ullman, J. D., Introduction to Automata Theory, *Languages and Computation*, 1979
2. Lande, R., Models of speciation by sexual selection on polygenic characters, *Proceedings of the National Academy of Sciences*, **78**, pp.3721-3725, 1981

3. West, J. M. and King, A. P., Female visual displays affect the development of male song in the cowbird *Nature* **334**, pp.244-246, 1988
4. Miller, G.F. and Todd, P. M. Evolutionary wanderlust: Sexual selection with directional mate preference, *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pp.21-30, 1993
5. Suzuki, J. and Kaneko, K., Physica D, **75**, pp.328-342, 1994
6. Lindgren, K. and Nordahl, M. G., Evolutionary dynamics of spatial games, *Physica D*, **75**, pp.292-309, 1994.
7. Zahavi, A. and Zahavi, A. *The Handicap Principle*, 1997
8. Werner, G. M. and Todd, P. M., Too Many Love Songs: Sexual Selection and the Evolution of Communication, *Advances in Artificial Life, 4th European Conference on Artificial Life*, pp.434-443, 1997
9. Honda, E. and Okanoya, K., Acoustical and Syntactical Comparisons Between Songs of the White-Backed Munia(Lonchura striata) and its Domesticated Strain, the Bengalese Finch(Lonchura striata var. domestica), *Zoological Science* pp.319-326, 1999
10. Lerena, p. Sexual Preferences: Dimension and Complexity, *Proceedings of the 6th International Conference of The Society for Adaptive Behavior, Appeared in From Animals to animats*, **6**, pp.395-404, 2000
11. Miller, G. F., Evolution of human music through sexual selection, *The origins of music* pp.329-360, 2000
12. Reggia J. A., Schulz R., Wilkinson G. S. and Uriagereka J., k Conditions Enabling the Evolution of Inter-Agent Signaling in an Artificial World, *Artificial Life*, pp.3-32, 2001
13. Okanoya, K., Sexual Display as a Syntactical Vehicle, *The Transition to Language*, pp.46-63, 2002
14. Sasahara, K. and Ikegami, T., Coevolution of birdsong grammar without Imitation *Advances in Artificial Life, 7th European Conference on Artificial Life*, pp.482-490, 2003
15. Sasahara, K. and Ikegami, T., Song Grammars as Complex Sexual Displays, *Artificial Life IX: Proceedings of the 9th International Conference on the Simulation and Synthesis of Living Systems*, pp.194-199, 2004
16. Lachlan, R. F. and Janik, V. M. and Slater, P .J. B., The evolution of conformity-enforcing behavior in cultural communication systems, *Animal Behavior*, pp.561-570, 2004
17. Okanoya, K., Song Syntax in Bengalese Finches: Proximate and Ultimate Analyses, *Advances in the Study of Behavior*, pp.297-346, 2004

A Fitness-Landscape for the Evolution of Uptake Signal Sequences on Bacterial DNA

Dominique Chu and Jonathan Rowe

School of Computer Science, University of Birmingham,
Birmingham B15 2TT, United Kingdom
{D.Chu, J.E.Rowe}@cs.bham.ac.uk

Abstract. In a recent article Chu *et al.* presented a computational model investigating the evolutionary origin of so-called *uptake signal sequences* in bacteria. In that contribution the authors used an agent-based approach. The main aim of this article is to understand the fitness-landscape on which the agents operate. We propose such a fitness-landscape and discuss its implications. This opens the possibility to use GAs for future simulations rather than the computationally expensive agent-based model.

1 Introduction

Many species of bacteria are, at least under certain conditions, *competent* meaning they have the ability to take up (relatively short) DNA fragments from their environment through genetically programmed developmental pathways[7,12,9]; natural competence is only one type of horizontal gene transfer available to bacteria. The others are *conjugation* and *transduction*. Those are not genetically controlled but are rather mediated by direct cell contact and phages respectively.

While many of the competent bacterial species are in-discriminant as to what type of DNA they take up, others have a strong preference for *conspecific* DNA fragments, i.e. DNA from dead members of their own species. Some of the species with this preference have a highly repeated uptake signal sequence (USS) on their DNA[10]. The USS mediates the uptake of DNA fragments from the environment. Examples of such species are *Haemophilus influenzae*, *Neisseria gonorrhoeae*, or *N. meningitidis*.

In *H. influenzae*[6,11] the USS is 9 bp long (`AAGTGCGGT`) and relatively evenly distributed throughout the DNA. Being repeated over 1400 times (on both strands) the USS is statistically highly overrepresented on the DNA; on a random DNA with the same G/C content one would only expect approximately 8 copies. About 34% of the USS are in non-coding parts of the genome (which makes about 10.4% of the DNA). Bakkali and coworkers[1] suggested that the remaining 56% of USS are contained in parts of genes that code for non-essential parts of the protein. USS in other species have similar statistical properties.

Researchers consider two scenarios[1,4,9] for the evolutionary origin of USS: The "**USS First**" scenario states that naturally competent bacteria had a

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 845–853, 2005.

certain preference to bind to USS. The high USS content is a result of recombinational inclusion of bound DNA fragments containing USS. This is opposed to the "**Preference First**" scenario according to which conspecific DNA is more beneficial than non-conspecific DNA. The USS evolved as a signal to allow bacteria to decide whether a DNA fragment stems from a member of their own species or not. Those bacteria that could effectively recognize conspecific DNA fragments had higher fitness.

Biologists have recently started to doubt that a USS can actually emerge in a "Preference First" scenario, arguing that this would require biologically unrealistic group selection[8]. In response to that Chu *et al.*[2,3] used a computational model to demonstrate experimentally that under the assumption of the "Preference First" scenario a USS actually emerges under a wide range of conditions (i.e. parameter settings).

Chu's model is an individual-based model with adaptive agents[5]. The fitness of agents is not evaluated according to a pre-defined fitness-function, but is an emergent result of the biologically motivated behavioral rules of the agents. However, in order to develop a clearer understanding of both the dynamics of the model and of the evolutionary challenges of real bacteria a clear understanding of the fitness-landscape for the evolution of uptake signals is essential.

In real biological systems it is often not possible to actually specify such a fitness-landscape. In this article we will argue that the emergence of USS is an exception in this respect. In section 4 we will specify a good approximation for a fitness-landscape capturing the essential features of the evolution of USS. The model itself will be described in section 2 and the results of one typical simulation run are presented in section 3. Finally, section 5 provides a discussion and a conclusion.

2 The Model

In this section we provide an informal overview of the basic elements of the computational model. The bacteria ("agents") live in an computational environment that contains strings of the letters `a,c,g,t` of length f (the *DNA fragments*). At every time-step the environment is replenished with *alien fragments*, i.e. randomly generated DNA fragments.

The agents themselves have a DNA of length $l \gg f$, i.e. strings of `a,c,g,t`. The main activity agents engage in is the uptake of DNA fragments from the environment. Before uptake, agents compare the candidate DNA fragment under consideration with a specific sub-sequence of their own DNA; the length of this subsequence is $u < u_{max}$, where u_{max} is determined by the user at compile-time.

Agents mainly die because of over-crowding. The environment in which agents live is assumed to have a limited carrying capacity. Once this upper limit is reached, for every new born agent, the oldest agent currently in the system will be killed. Another possible reason for the death of an agent is old age. Whenever an agent is killed, a randomly chosen piece of its DNA is placed into

the environment. These pieces taken from agents constitute the second type of DNA, which we will refer to as *bacterial fragments*.

Agents can give rise to offspring if they have collected enough pay-off. Newly born offspring might be mutated. There are several ways in which mutations can affect the agent: Firstly, the DNA itself might be changed (either through point mutations or copy-mutations). Secondly, mutations might change the length or position of the sub-sequence that determines the receptor sequence.

The following algorithm is used to update the agents:

1. Initialise the population of agents by endowing them with a randomly generated DNA of fixed length l.
2. Specify for each agent a random subsequence of the DNA with maximal length u_{max}; this subsequence functions as receptor sequence s.
3. Seed the environment with randomly generated DNA fragments of length f.
4. Update all agents as follows:
   (a) Check k DNA fragments from the environment for the sequence s.
      i. If no fragment contains s, then take the last fragment.
      ii. If a match is found, then take the matching fragment and abort search.
   (b) Exchange the fragment for payoff points. Randomly generated ("alien") fragments and fragments taken from dead agents ("bacterial") receive different amounts of payoff.
   (c) If the accrued number of payoff points exceeds a certain threshold then reproduce.
      i. If the system has reached its carrying capacity, then kill off one of the oldest agents to create space for offspring.
      ii. Offspring is mutated with probability *mut* in one of the following ways:
         – A point mutation of the DNA
         – A copy-mutation of the DNA
         – Change the subsequence of the DNA that is used to determine s in one of the following ways:
            • Shift it by one, either to the left or to the right
            • Adjust its length by ± 1 either to the left or to the right
   (d) If an agent has reached the maximum age, kill it.
5. For each agent killed in this time step choose a random piece of its DNA of length f and place it into the environment.
6. Refill the environment with randomly generated DNA fragments of length f.
7. Goto 4.

3 A Typical Simulation Run

An exhaustive discussion of experimental results under various parameter settings can be found in [2,3]. In this section we will only briefly describe one typical simulation run showing the essential features of the emergence of USS (see

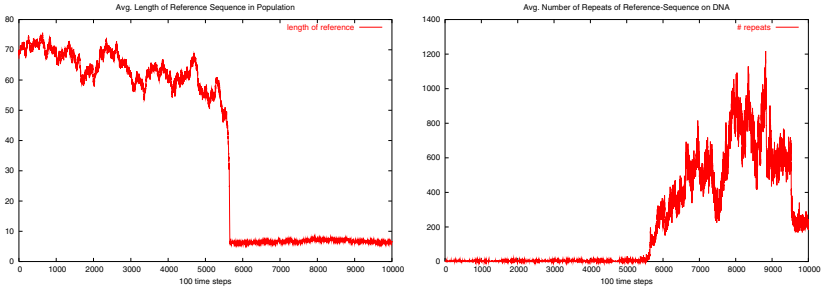


Fig. 1. Results from a typical simulation. We use the parameters in table 1. **Left:** Time series of the length of the reference averaged over the population. One measurement is taken every hundred time steps. **Right:** The number of repeats of the reference sequence on the DNA. Note that this measure is very noisy. Particularly, there exist some agents with very short reference sequences that cause considerable fluctuations of the average value of the repetition of the reference sequence.

figure 1; see table 1 for the parameters used). We found that the behavior of the model is best understood by looking at the behavior of the following variables over time:

- Length of the Reference: This variable measures the length of all the references in the population averaged over the population.
- Repetitions of Reference: Measures the number of repetitions of the reference sequence of an agent on her DNA averaged over the population.

An efficient USS must be neither too short nor too long, thus maximizing the agents' ability to find the USS on conspecific DNA fragments while minimizing false positive recognitions. Furthermore, in order to function effectively as an uptake signal, a USS needs to be highly repeated on the DNA. Thus, if a USS emerges in the current model, we would expect the reference sequence to be rather short, and also highly repeated on the DNA.

In figure 1 the emergence of a USS in the population is indicated by the discontinuity between time step 500000 and 600000; at this time the average reference length drops to a low value and the average number of repetitions of the reference sequence on the agents' DNA increases. Note that some of the repetitions of the reference sequence are due to very short reference sequences in the population. However, the emergence of a USS is also indicated by a sharp increase of the rate of correct recognitions of bacterial fragments (data not shown). We thus conclude that in this particular run the USS emerges.

4 The Reduced Fitness-Landscape

Agent-based models such as the present one do not assume a pre-specified fitness-landscape (for example a fitness function), but rely on *implicit* fitness that emerge

from the actions and interaction rules of the agents. Those actions/interactions are in turn motivated by the problem at hand (in this case the evolution of USS mediated uptake). Particularly if the interactions between agents is relevant for their reproductive success, then it might not be possible to describe a relevant fitness-landscape for the particular model. In the present case, however, there is no direct interaction between the agents. A complication of the current model is that at each time the fitness of a population partly depends on the DNA of past populations, as fragments taken from them serve as food for the current population. This might effectively prevent us from finding a general description of the fitness-landscape. However, if we assume that the DNA of the population changes only slowly compared to generation times, then we can also assume that past generations (and thus the bacterial fragments that are currently in the environment) will be very similar to the current population. A possible measure of the fitness of an agent is thus the probability with which she would find her reference sequence on a randomly taken piece of her own DNA of the length of a food fragment, while at the same time avoiding false positive recognitions. This measure is not a completely accurate description, but it provides nevertheless some important insights about the evolutionary pressures the agents face.

Table 1. The values of the parameters used in the simulation runs presented in section 3

DNA length	10000
World Size	30
population max. size	300
mutation Rate	0.9
energy for bacterial fragment	3
max. number of fragments presented to agents	20
size of fragment	100
min. energy to reproduce	6
maximum lifetime	10
limit lifetime	20

Conventionally fitness-landscapes are considered over the space of possible genetic configurations; in the particular case there is another useful representation. In the current model the fitness of agents primarily depends on the reference sequence and how often it is repeated on the DNA, whereas the particular letter sequence of the DNA is irrelevant. It is therefore instructive to consider a reduced (and easy to represent) fitness-landscape first in order to generate an understanding of the adaptive pressures involved.

Let the length of the DNA be l, the fragment size f, the length of the reference sequence u and the number of repetitions of the reference sequence on the DNA n. Then the probability that a random fragment of length f taken from a DNA will contain a specific signal is equal to the probability that at least one of the last $f - u$ letters of the fragment contains the last letter of a copy of the reference sequence.

$$P_1 = 1 - \left(1 - \frac{n}{l}\right)^{f-u+1}$$

In order to calculate the fitness it is also important to take into account the possible influence of false positive recognitions. In order to estimate this we need to know the probability that a random sequence of length f contains the specific

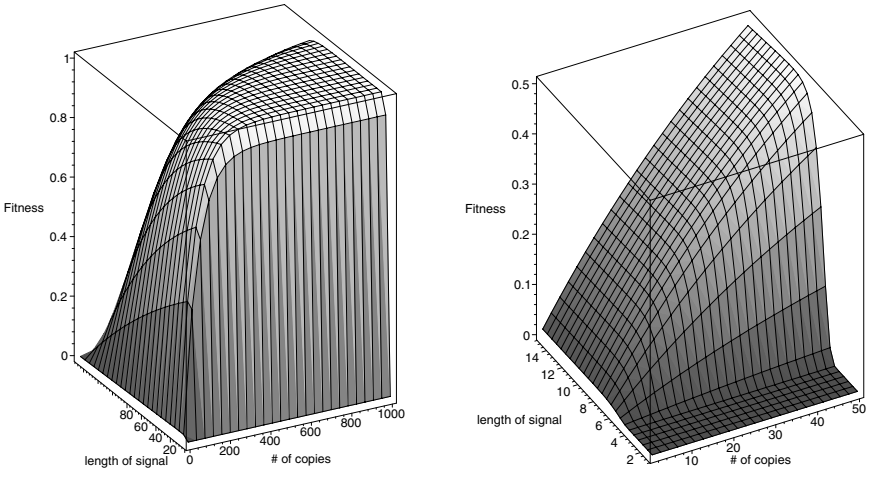


Fig. 2. The reduced fitness-landscape over the number of copies of a signal and the signal length (according to eq. 4). For this particular graph we assume a fragment size of 150 and DNA length of 10000. The right figure is a blow-up of the left figure. Note that agents will be very restricted in the possible paths to climb the fitness-landscape. See main text for an explanation.

reference sequence of length u. The probability that a random sequence contains at least one such sub-sequence is equal to the probability that it does **not** contain **none**. This probability can be calculated by using a simple sliding window method. The first window covers the first to the u-th letter of the fragment. The probability that this window is equal to the reference sequence is $1 - (1/4)^u$. The second window covers the second to the $(u+1)$-st position of the fragment and has the same probability not to contain the reference sequence. Altogether, we have to consider $f - u + 1$ windows to make sure that the entire fragment does not contain any reference sequences. We thus obtain the probability that a random fragment contains at least one copy of the reference sequence.

$$P_2 = 1 - \left(1 - \left(\frac{1}{4}\right)^u\right)^{f-u+1}$$

Having calculated those probabilities, we can now give an estimate for the reduced fitness of an agent, based on the observation that an agent is fitter if it is efficient in recognizing fragments taken from its own (or very similar) DNA sequences without mistaking random sequences for bacterial fragments.

$$Fit = P_1(1 - P_2) \tag{1}$$

While this reduced fitness-landscape allows us to calculate the associated fitness of an agent given its DNA and GENOME, it does not allow us to directly show what fraction of the possible genomic configurations correspond to a particular

fitness-value; another drawback is that it somewhat misrepresents the neighborhood relations of points as not all points that are close to each other on the reduced fitness-landscape are also close to each other in the un-reduced fitness-landscape over the genomic space. To see this, consider as an example the case of a DNA of length 10 and a signal of size 6. If the DNA is homogeneous, that is its entire sequence consists of the same letters, then there will be 5 copies of the signal on the DNA. No mutation will be able to take the agent to the neighboring point in the reduced fitness-landscape with 6 copies of the signal and a reference length of 6 as this does not correspond to any possible genomic configuration. On the other hand, a point-mutation within the reference sequence will immediately take the agent to a place on the reduced fitness-landscape corresponding to a reference sequence of length 6 and one copy of it. Thus, while the proposed measure of fitness does allow us to calculate the fitness of any given agent, it does not truthfully reflect neighborhood relations between possible genomic configurations of agents.

In all simulations agents are initialized with random genomic configurations (that is the DNA and the reference sequences will be random). Consequently, the reference sequence will take a random value from one to the maximally allowed value. The probability that the reference sequence is short (say <12) is 12 divided by the maximum allowed size of the reference sequence; for high maximum allowed values it might thus be relatively small. The probability that a random DNA contains two copies of a longer reference sequence is vanishingly small for all DNA lengths we simulated. Thus, initially most agents are in the the one-dimensional sub-space $n = 1$ (compare figure 2) of the fitness-landscape; in figure 1 this corresponds to few repetitions of the reference sequence before the emergence of the USS. As can be seen from figure 2 in this part of the fitness-landscape the fitness is very low.

From there, there are two ways to go: Firstly, copy-mutations might increase the number of copies of the reference sequence. The probability P_3 that a copy-mutation creates a new copy of the signal is the probability that the source of the copy-mutation contains at least one copy of the relevant sequence and that the target does not contain any. If the length of the copy-mutation is c, then the former probability is equal to the probability that the last $c - u + 1$ letters of the source area of the copy-mutation contain the last letter of one of the n repetitions of the reference signal; similarly, the probability that the target does not contain any letter of a copy of the reference sequence is the probability that none of the letters of the target area and none of the u letters to the right of the target area contain a last letter of the reference sequence.

$$P_3 = \left[1 - \left(1 - \frac{n}{L}\right)^{c-u+1}\right] \cdot \left(1 - \frac{n}{L}\right)^{c+u-1} \tag{2}$$

The probability that a copy-mutation produces a new copy of the reference sequence depends on the length of the reference sequence. The longer the reference sequence the more unlikely it becomes that a copy-mutation creates a second copy of the reference sequence; moreover, even if there is a copy of the reference sequence, the probability of it being destroyed by a copy-mutation increases with

its length. For agents it is thus very difficult to leave the sub-space $n = 1$ as long as their reference sequence is very long. Even if they leave this sub-space, they are then likely to get stuck in the sub-space $n = 2$.

Agents can go in another direction. Mutations of the GENOME can change the length of the reference-sequence. As can be seen by looking at the derivative of the fitness-function in the sub-space $n = 1$ (see figure 3), the slope of the fitness function is essentially zero for longer (>12) reference sequences. In this context it is interesting to note that the fitness is maximal (i.e. its derivative is zero) for a reference length of about 9, closely corresponding to the length of the USS in *H. influenzae*. This means that all mutations of the GENOME in this area are essentially neutral. Strong adaptive pressures will only appear for short reference sequences. Thus, while the reference sequences are very long the agents essentially perform a one-dimensional random walk over the space of possible reference sequence lengths. The average displacement from the starting point in such random walks is zero, but the average distance traveled after N steps is proportional to $N^{\frac{1}{2}}$. Note that N does not correspond to actual time-steps in the model, but to mutation events changing the length of the reference sequence. If the mutation rate is m, the point mutation rate is p, the average age at which an agent produces offspring is t, and the population size is s, then the reference sequence will be adjusted only every $8t/m(1-p)$ time steps. In order to observe the emergence of USS in the model, it is thus crucial to restrict the initial size of the reference sequence.

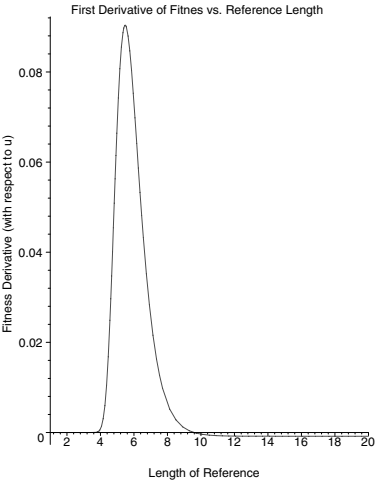


Fig. 3. This figure shows the derivative of the fitness with respect to a change of the reference length, when n is set to one. This shows that for reference lengths >10 changes of the reference length have a very small relevance for the fitness.

5 Discussion and Conclusion

From this we can now understand how the USS emerges in the model. Initially agents will drift through the sub-space $n = 1$ without being subject to a strong adaptive gradient. In figure 1 this corresponds to the first half of the run, where the length of the (average) reference sequence is between 50 and 70 long and there are very few copies of the reference sequence on the DNA.

The sudden drop indicates the point where one or several agents have managed to reduce the length of their reference sequence sufficiently; a reference sequence of about 9 or 10 optimizes the probability to recognize conspecific fragments, while avoiding false-positive recognitions. From then on they are driven upward on the fitness-landscape by increasing the number of copies of the reference sequence on their DNA.

Acknowledgments

We are grateful to the Paul & Yuanbi Ramsay Research Fund for generous support of this work.

References

1. M. Bakkali, T. Chen, H. Lee, and R. Redfield. Evolutionary Stability of DNA Uptake Signal Sequences in the Pasteurelleceae. *Proceedings of the National Academy of Science*, 101(13):4513–4518, 2004.
2. D. Chu, H Lee, and T. Lenaerts. Emergence of Uptake Signals in Bacterial DNA. *Artificial Life*, 11(3), 2005. forthcoming.
3. D. Chu, J. Rowe, and H. Lee. Models of the Evolution of Uptake Signal Sequences. *Journal of Theoretical Biology*, 2005. accepted.
4. T. Johnson and R. Redfield. The Molecular Evolution of DNA Uptake Signal Sequences, 2003. in preparation.
5. O. Judson. The Rise of Individual Based Models in Ecology. *Tree*, 9(1):9–14, 1994.
6. S. Karlin, J. Mrazek, and M. Campell. Frequent Oligonucleotides and Peptides in the *Haemophilus Influenzae* Genome. *Nucleic Acids Research*, 24:4263–4272, 1996.
7. M. Lorenz and W. Wackernagel. Bacterial Gene Transfer by Natural Genetic Transformation in the Environment. *Microbiology and Molecular Biology Reviews*, 58(3):563–602, 1994.
8. R. Redfield, 2001. NIH proposal to study the evolution and function of uptake signal sequences in naturally competent bacteria, available at: http://www.zoology.ubc.ca/ redfield/research/USS/USSprpsl.pdf.
9. R. Redfield. Do Bacteria Have Sex? *Nature Review Genetics*, 2:634–639, 2001.
10. H. Smith, M. Gwinn, and S. Salzberg. DNA Uptake Signal Sequences in Naturally Transformable Bacteria. *Research in Microbiology*, 150:603–616, 1999.
11. H. Smith, J. Tomb, B. Dougherty, R. Fleischmann, and J. Venter. Frequency and Distribution of DNA uptake Signal Sequencs in the *Haemophilus Influenzae* Rd Genome. *Science*, 269:538–540, 1995.
12. J. Solomon and A. Grossman. Who's Competent When: Regulation of Natural Genetic Competence in Bacteria. *Proceeedings of the National Academy of Science (PNAS)*, 97:6981–6985, 2000.

The Genetic Coding Style of Digital Organisms

Philip Gerlee and Torbjörn Lundh

Department of Mathematical Sciences, Chalmers University of Technology,
SE-412 96 Göteborg, Sweden
http://www.math.chalmers.se/~torbjrn/coding_style

Abstract. Recently, all the human genes were identified. But understanding the functions coded in the genes is of course a much harder problem. We are used to view DNA as some sort of a computer code, but there are striking differences. For example, by using entropy, it has been shown that the DNA code is much closer to random code than written text, which in turn is less ordered than ordinary computer code. Instead of saying that the DNA is badly written, using common programming standards, we might say that it is written in a different style – an evolutionary style. In this paper the coding style of creatures from the artificial life platform Avida has been studied. Avida creatures that have evolved under different size merit methods and mutation rates have been analysed using the notion of stylistic measures. The analysis has shown that the evolutionary coding style depends on the environment in which the code evolved, and that the choice of size merit method and mutation probabilities affect different stylistic properties of the genome. A better understanding of Avida's coding style, might eventually lead to insights of evolutionary codes in general.

1 Introduction

It was shown, using block entropy, in [1], that the DNA is much closer to random code than human written computer code. Furthermore a lot of examples have been found where genes have been reused for different purposes during development. As an example, take the **runt** gene in *Drosophila*, which has been shown [2] to be used in sex determination, segmentation, and central nervous system creation. These findings suggest that the DNA is coded in a rather different fashion compared to human computer code.

C. Adami made in his survey talk in Stony Brook 10/27/98 on artificial life, a brief remark about the quality of the evolved program codes of the digital organisms [3] in the artificial life platform Avida [4]:``*The codes that are evolved will eventually be almost totally unreadable. Things are never used only once, but two or more times. It is a kind of a 'madman's' code.''*

How can we capture these comments about the style, or quality, of the computer code and the DNA, in a quantified manner? Can we do that in such a general manner that we will be able to use analogous quality measure both for carbon – and silicon based genetic codes?

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 854 – 863, 2005.

When a population evolves and adapts to an environment, information about what traits or chemical reactions that are beneficial in that environment are written into the genome. The environment of course influences *what* information is coded into the genome, but what we are interested in is to investigate *how* the information is coded. In this paper we will investigate the different coding styles that the environment enforces on the genomes by examining Avida creatures that have evolved under different size merit methods and mutation rates. This will be done using the notion of a stylistic measure that was introduced in [5].

2 Codes

One common way to view functions is as black boxes. Here we are interested in the internal structure of such black boxes performing equivalent tasks. Let us give a simple example of two codes that interprets the same simple real valued function:

$$f(x) = \frac{1}{x-2} \qquad g(x) = \begin{cases} \dfrac{x+2}{x^2-4} & \text{if } x \neq -2 \\ -\dfrac{1}{4} & \text{if } x = -2. \end{cases} \tag{1}$$

The genome of a creature can be thought of as a code that is written in an alphabet consisting of a finite numbers of letters, which is read or interpreted by the CPU. The genomes found in nature are written with the four letters A, T, G and C, which corresponds to the base pairs in the DNA. In Avida the genomes consists of a combination of 24 different CPU instructions, which alter the state of the virtual CPU in some manner.

We will therefore define a **code** to be a finite string of letters taken from a finite alphabet A, such that when the code is interpreted, the code will represent a well-defined function or process, f.

$$Code = \{\alpha_i\}_{i=1}^{k}, \quad \text{where } \alpha_i \in A. \tag{2}$$

From the above definition it is clear that different codes can have the same function representation. Let us therefore define a class of codes C_f to be the set of all codes that perform the function f when they are interpreted. From a genetic point of view one can interpret the function of the code as the phenotype of a creature, and thus the code classes as classes of phenotypically equivalent creatures.

In Avida the interpretation is performed by running a creature through the virtual CPU for one generation. If a successful divide occurs the function that the code defines is simply the tasks that the creature performs during one generation.

3 Styles of a Code

If we look at two codes from the same class we know that they perform the same function, i.e. they have the same phenotype, but their genotype may differ. As the

evolutionary process is very sensitive to perturbations we cannot expect to find the same genotype if evolution occurred two times in the same environment. It's therefore interesting to study the style of the code, as this is more likely to be invariant between two instances of the same process. We would therefore like to define the style of a code. This is done by introducing measures (3) on the code class C_f, that maps each code to a point in [0,1].

$$\mu_i : C_f \rightarrow [0,1]. \tag{3}$$

Putting together several of these measures we can create a profile measure $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3, \ldots)$, which might serve as a 'fingerprint' of the code.

4 Measures

As we are interested in distinguishing between different coding styles of creatures from Avida we have constructed four different measures that measure different properties of the genome. Three of the measures are calculated from the functional genomic array (FGA) of the genome, a representation which reveals the genetic structure and the localisation of genes [6]. The FGA is a $N \times M$ binary matrix, where N is length of the genome and M is the number of tasks the creature manages including replication. The entry at position (i,j) is 1 if instruction number i is involved in the calculation of task number j and 0 otherwise. An example of a typical FGA can be found in fig. 3.

4.1 Gene Correlation

This measure shows how correlated different genes are and is constructed in the following manner. Sum the functional genomic array along the rows, and remove all zero entries in the resulting vector. Sum up all entries that are larger than one and divide by the number of tasks plus one (for replication) and by the number of non-zero entries in the vector.

If we let $\mathbf{T}$ be the row sum of the FGA, N be the number of tasks the creature manages plus one (for replication), L_E, the number of non-zero entries in $\mathbf{T}$ (the number of essential instructions), then the gene correlation is given by (4), where the sum runs over all indices for which $\mathbf{T_i} > 1$.

$$g_c = \frac{\sum_i \mathbf{T}_i}{NL_E}. \tag{4}$$

The gene correlation is a number between 0 and 1, where 0 means that all tasks including replication are coded disjoint in the genome, and 1 means that they all depend on the same instructions. Note that this measure also can be interpreted as the compression of information in the genome.

4.2 Redundancy

This measure gives the fraction of instructions that don't affect the tasks and replication of a creature. As above the FGA is summed along the rows, but this time we count how many zero entries the vector holds, this number is then normalised by the length of the creature.

4.3 Introns

This measure simply gives the fraction of instructions that never are executed when the creature is executed for one life-cycle. Under the default size merit method 4, the fraction of introns is rather low (see table 1), as they are punished. Note that the introns are a subset of the redundant instructions.

4.4 Fragility

This measure gives the number of instructions that are essential for replication, and is constructed from the FGA by simply summing the replication column. This measure isn't normalised with the length of the creature because the number of instructions needed to replicate are independent of the length of the creature, as the creatures use copy loops. But as we want a measure to lie in [0,1] this measure is normalised using the map (5).

$$f(x) = \frac{x}{x+c}, \quad c > 0 \tag{5}$$

From preliminary data we know that the number of instructions essential for replication in most cases lie between 5 and 50. The c in (5) is therefore optimised so that the image of the interval $f([5,50])$ is maximised. The optimisation is straight forward and gives the value $c = \sqrt{(50 \cdot 5)} \approx 15.8$, which gives $f(50) \approx 0.76$ and $f(5) \approx 0.24$. In an environment that doesn't reward computational efforts it is only the fragility that is minimised [7], which results in a minimised genome length.

5 Comparison of Different Styles

5.1 Size Merit Methods

The different size merit methods in Avida generate qualitatively different styles of coding, for example size merit method 1, which gives merit proportional to copied size, tends to produce introns and does not put any pressure on the lengths of the creatures, while size merit method 0 does quite the opposite, because merit is independent of size. The third size merit method, 4, takes the minimum of copied and executed size as merit.

The merit is a very important property as it is used in the calculation of the fitness. This implies that the size merit method has a direct impact on how selection is performed in Avida [4].

The question is if these differences in merit calculation can show in a stylistic analysis of the different methods. To investigate this we created three sets of creatures each containing approximately 60 creatures, from the three size merit methods 0, 1, 4, the full settings for these runs can be found in the appendix.

The optimal comparison would be to compare creatures from the same code class (phenotype), but as evolution in Avida proceeds with different pace each run one has no guarantee that a certain phenotype has evolved after a fixed number of updates. One way to accomplish this would be to reward only those functions that we want the phenotype to perform and use a large number of fixed updates, but this approach also has its drawbacks. If the required phenotype appears early in evolution, then the code is optimised during the rest of the run as no new functions are rewarded. If a creature from the above run is compared to a creature from a run where the required phenotype appeared just before the maximal number of updates their coding styles would certainly differ, as one creature has optimised its coding while the other has not. Instead we decided to compare creatures with approximately the same complexity. This was done by extracting a creature from the dominant genotype after 40 000 updates in each run and if it had reached a certain degree of complexity (it managed at least three boolean functions) it was kept for analysis.

Table 1 shows the average and standard deviation for each measure under the three size merit methods. Another way to analyse the data is to perform a principal component analysis on the data, which reduces the dimensionality to 2, and gives a better graphical representation. The result of the PCA can be seen in fig. 2. The vectors that span the plane in fig. 1, are the 1st and 2nd principal components (6).

Table 1. The average and standard deviation of the stylistic measures for creatures from three different size merit method

Size-Merit Method	Gene correlation	Redundancy	Introns	Fragility
0	0.2655 (0.1221)	0.1666 (0.1102)	0.0249 (0.0496)	0.4860 (0.0402)
1	0.2647 (0.1161)	0.6443 (0.1836)	0.3581 (0.1767)	0.5639 (0.0706)
4	0.2288 (0.0990)	0.5230 (0.1905)	0.0708 (0.0911)	0.5565 (0.0705)

$$\begin{aligned} P_1 &= \begin{pmatrix} -0.0132 & -0.6263 & -0.5723 & 0.1700 & -0.5011 \end{pmatrix} \\ P_2 &= \begin{pmatrix} 0.7861 & -0.1275 & 0.0156 & -0.5990 & -0.0825 \end{pmatrix} \end{aligned} \tag{6}$$

5.2 Mutation Rates

The mutation rates in Avida play an important role in the adaptive process. If the mutation rates are low evolution tends to proceed very slow as the fitness landscape is explored at a low pace and if they are high the population will have trouble sustaining information in the genomes [8,9].

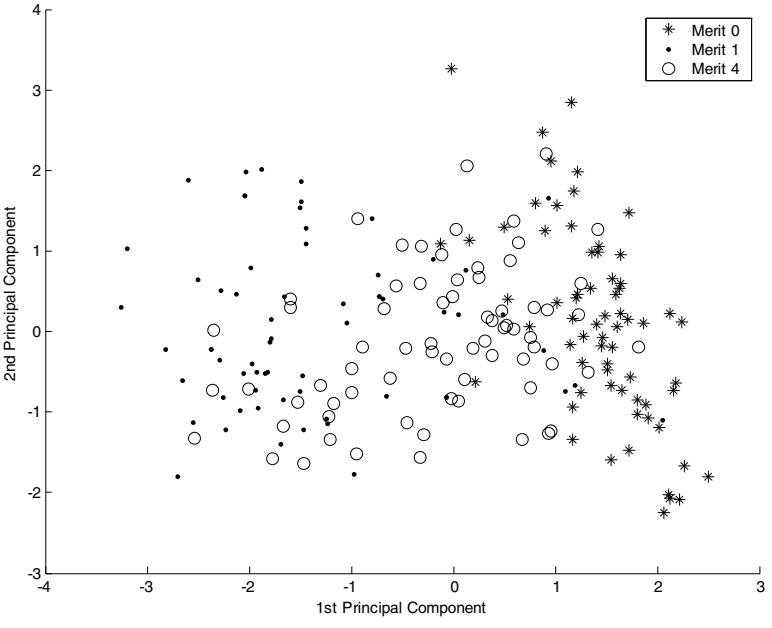


Fig. 1. Principal component analysis of the profile measure of creatures that have evolved under different size merit methods

To investigate how the mutation rates influence the coding style we created three sets of creatures that had evolved under different copy mutation probabilities each containing approximately 60 creatures. The point mutation rates were set to zero, the insert/delete mutation probabilities were kept constant at 0.05 per divide and the size merit method was set to 4 (the default value in Avida). The copy mutation probability was set to one low value (p_c = 0.001), one intermediate value (p_c = 0.005) and one high value (p_c = 0.025).

Table 2. The average and standard deviation of the stylistic measures for three sets of creatures that have evolved under different copy mutation rates

p_c	Gene correlation	Redundancy	Introns	Fragility
0.001	0.2263 (0.1093)	0.6076 (0.2006)	0.0487 (0.0883)	0.5877 (0.0811)
0.005	0.2288 (0.0990)	0.5230 (0.1905)	0.0708 (0.0911)	0.5565 (0.0705)
0.025	0.3037 (0.1372)	0.2847 (0.1558)	0.0372 (0.0985)	0.4872 (0.0465)

As in the experiment above the populations evolved for 40 000 updates after which the dominant genotype was extracted, but it was only kept for analysis if it managed three or more boolean functions. A detailed description of the settings can be found in

the appendix. The three sets of creatures where then analysed using the stylistic measures and the results can be found in table 2 and a PCA-plot in fig. 2. The 1st and 2nd principal components are given by (7).

$$P_1 = \begin{pmatrix} -0.2087 & 0.6374 & 0.4462 & 0.2359 & 0.5435 \end{pmatrix}$$
$$P_2 = \begin{pmatrix} -0.7657 & -0.0693 & -0.5643 & 0.2695 & 0.1337 \end{pmatrix} \quad (7)$$

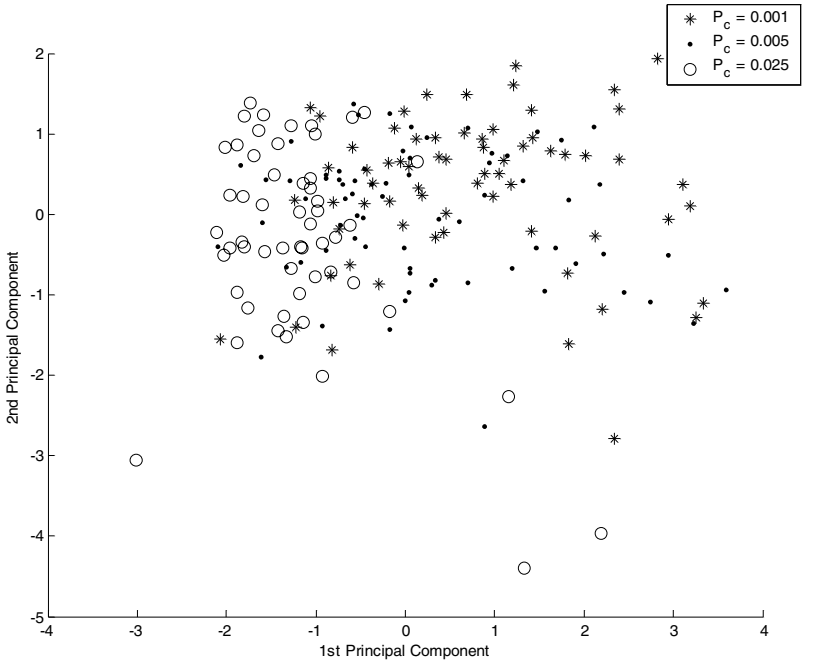


Fig. 2. Principal component analysis of the profile measure of creatures that have evolved under different copy mutation probabilities

6 Discussion

6.1 Size Merit Methods

In fig. 1 one can see a separation between the different size merit methods, which clearly indicates that there are differences in their coding style enforced by the size merit method. One way of analysing these differences is to look at the composition of the 1st principle component (6). The measures with the larger weights show where the styles differs the most. This shows that the coding styles differ most with respect to the redundancy, intron and fragility measures. What can also be seen in the PCA plot is that the coding style within each size merit methods varies quite much, a thing that also can be seen in the standard deviations in table 1. The reason for this is that the evolutionary process in Avida takes different paths each run due to the randomness in

the process, and thus giving rise to a unique coding each run. The large standard deviations makes it impossible to draw any conclusions about how the gene correlation is affected by the size merit method, but the other measures show separation between the different size merit methods.

The results of this experiment are intuitive and can be explained directly from how the merit is calculated. Size merit method 1 for example has a high fraction of introns as the method gives merit proportional to copied size and therefore does not punish introns and method 0, which gives merit independent of size, gives a efficient coding with a low fraction of both introns and redundant instructions.

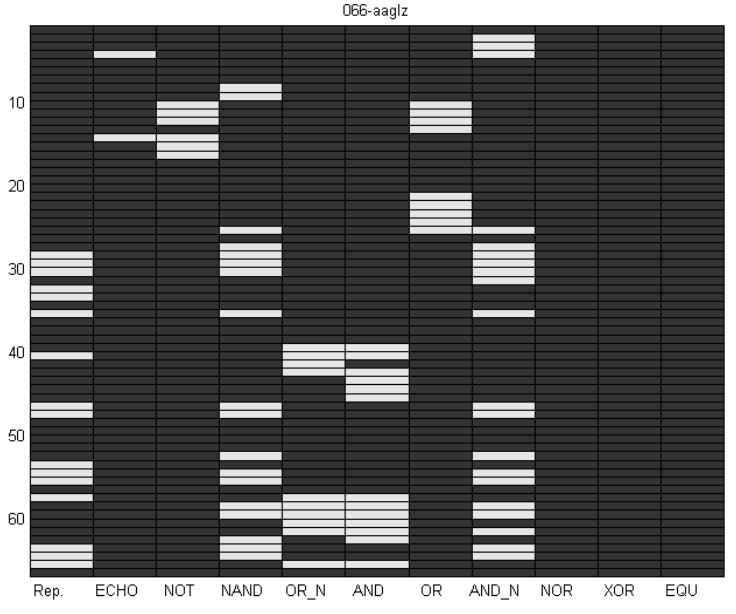


Fig. 3. The Functional Genomic Array (FGA) of a typical Avida creature. Each column in this array indicates a task, including replication, where the line element is white if the task depends on that specific line in the Avida code.

6.2 Mutation Rates

The principal component analysis plot (fig. 2) in this case does not show the same separation between the coding styles as in the case with size merit method. The coding style of the creatures from the low and intermediate mutation probabilities seem clustered together, but the coding style of the high mutation probability show at least some separation from the other two. The 1st principal component (7) shows that the largest differences between the coding styles lie in the redundancy, fragility and gene correlation measures.

The averages of the stylistic measures (table 2) shows the same tendency as the PCA plot, the low and intermediate mutation probabilities give approximately the

same values while the high probability differs in the gene correlation, redundancy and fragility measures.

The fragility decreases when the mutation probability increases. The reason for this is that a high mutation probability requires a more efficient coding of the self-replication. A creature that uses too many instructions for self-replication would be less likely to produce a viable offspring when the copy mutation probability is high.

The gene correlation on the other hand increases when the mutation probability increases. The high mutation probability forces the creatures to compress the information in the genome, in order to make it less likely to be struck by a deleterious mutation. This compression corresponds to that the genes share instructions which gives a higher gene correlation.

The reason why the redundancy decreases when the mutation probability increases is because a redundant instruction that is copied incorrectly may alter the execution in the offspring which may lead to the loss of a gene or even the capability to self-replicate. It is therefore disadvantageous to have a high redundancy when the mutation probability is high.

7 Conclusion

The results of the experiments clearly show that settings in Avida produce different coding styles. Most of the differences that appear are intuitive and can be explained directly from for example how the merit is calculated. While some results, like the change in gene correlation in the experiments with mutation probabilities, requires an understanding of the evolutionary process in Avida.

What these experiments show is that different environmental settings affect different stylistic properties of the genome. The gene correlation does not seem to depend much on the size merit method but rather on the mutation probabilities and the fraction of introns seems independent of the mutation probabilities but depends strongly on the size merit method. The fragility measure on the other hand seems to depend on both mutation probabilities and size merit method. But the main result is that we can distinguish between different coding styles from different environments using a stylistic profile measure. It would be very interesting if one could study other evolutionary driven systems using this stylistic approach in order to look for universal features of evolutionary driven code in general and DNA in particular.

References

1. Schmitt, A.O., Herzel H.: Estimating the entropy of DNA sequences, Journal of Theoretical Biology, 188: 3, (1997), 369-377
2. Duffy, J., Gergen, P.: Sex, Segments, and the Central Nervous System: Common genetic mechanisms of cell fate determination, Adv. in Genetics, Vol. 31, (1994) 1-28
3. Wilke, C.O., Adami, C.: The biology of digital organisms. Trends. Ecol. Evol., 17, (2002), 528-532

4. Ofria C., Wilke, C.O.: Avida: A Software Platform for Research in Computational Evolutionary Biology. Artificial Life, 10, (2004), 191-229
5. Lundh, T.: In search of an evolutionary coding style, SUNY Stony Brook IMS preprint 3, (2000)
6. Lenski, R.E., Ofria C., Penncock, R.T., Adami, C.: The evolutionary origin of complex features, Nature 423, (2003) , 139 - 144
7. Lenski, R.E., Ofria C., Collier, T.C., Adami, C.: Genome complexity, robustness and gene interactions in digital organisms, Nature 400, (1999), 661 – 664
8. Adami, C., Edlund, J.A.: Evolution of robustness in digital organisms. Artificial Life 10, (2004), 167-179
9. Wilke, C.O, Wang, J.L., Ofria, C., Lenski, R.E., Adami, C.: Evolution of digital organisms at high mutation rate leads to survival of the flattest. Nature, 412, (2001), 331-333

Appendix

All experiments for this paper were performed with Avida 1.3.0 for Windows. The settings used for the experiments were the default settings in Avida, except for the changes in size merit method and copy mutation rates. The task bonuses were set to the default value except for the fact that the rewards for 3-input boolean functions were removed. In the runs with size merit 0 the task bonuses were raised slightly in order to prevent the evolution from going in to a size minimising state. Each run was started with a time based random seed and the ancestor used in all experiments was `creature.base`, which is supplied with Avida. The ancestor, `genesis` and `task_set` files for all experiments can be downloaded from http://www.math.chalmers.se/~torbjrn/coding_style.

Growing Biochemical Networks: Identifying the Intrinsic Properties

Hugues Bersini, Tom Lenaerts, and Francisco C. Santos

IRIDIA, CP 194/6,
Université Libre de Bruxelles, Brussels, Belgium
bersini@ulb.ac.be

Abstract. How can a new incoming biological node measure the degree of nodes already present in a network and thus decide, on the basis of this counting, to preferentially connect with the more connected ones? Although such explicit comparison and choice is quite plausible in the case of man-made networks, like Internet, leading the network to a scale-free topology, it is much harder to conceive for biochemical networks. The computer simulations presented in this article try to respect simple and, as far as possible, basic biological characteristics such as the heterogeneity of biological nodes, the existence of natural hubs, the way nodes bind by mutual affinity, the significance of type-based network as compared with instance-based one and the consequent importance of the nodes concentration to the selection of the partners of the incoming nodes.

1 Introduction

Recent years have brought a resurgent interest for evolving networks showing interesting and far from random connectivity structure like a power-law or scale-free one (Barabási and Albert) (BA in the following) [1,2,3,4,5,6,7]. Although the most representative of these networks have been spotted in the human and social worlds (like the Internet or epidemic networks), the fact that such a connectivity structure allows the nodes to optimally connect (these networks exhibit the small-world and good robustness properties allowing a fast and reliable communication), has encouraged an increasing number of biological researchers to believe that this scale-free connectivity should hopefully be shared by biological networks. Based on rough experimental data, some cellular, genetic and chemical networks seem, as a matter of fact, to structure their connectivity in such a particular way [8,9,10,11,12,13].

In this paper, computer experiments of growing networks will be proposed as more elementary and tractable versions of a long tradition of simulations of immune networks and chemical reaction networks popular in Artificial Life. When adopting a more biological perspective, the BA preferential attachment [1] poses

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 864–873, 2005.

serious difficulties[1]. How could a new biological node, discovering and observing *potential partners*, preferentially decide to connect with one of these on the basis of its connectivity? Although a human being can perform such a conscious choice while involved in the construction of any technological network (such as the Internet, the Web or public transportation) or entering a sexual network, namely to express a certain preference for some nodes to attach to, this same preferential choice appears quite unlikely in natural systems growing with no human intervention. In order to remain faithful to observed biological networks, some basic principles need to be incorporated: (i) every node has a different identity based on its physical properties defining its *type*; (ii) every node connects to a selected set of nodes based on mutual *attractiveness* (*affinity*) ; (iii) certain nodes have intrinsically more ways to connect than others i.e. they are *natural hubs* and (iv), since every node represents a type, biological networks are *type-based* network instead of *instance-based*.

Take for instance chemical [15] or metabolic networks [9]. Here molecules appear only once in the network and the connections refer to the reactions in which these molecules are involved. Every node corresponds to a molecular type and has a particular concentration, and this will play a key role in the attachment mechanism: a new node should connect to other nodes based on the distribution of concentrations since this determines the probability of interaction. Furthermore, it is well-known that some molecules are more reactive than others due to their structure. Consequently, some molecules are more attractive than others. Due to this, the connectivity of a node is not only an outcome of the growing history (like in [1]), but also a result of the nodes' intrinsic features. Some of these nodes have such a high attractiveness that they become hubs (in protein networks, p53 is famous for that [16]) prior to any connection with others; they were born like hubs. Given this interpretation of molecules, chemical reaction networks represent interactions between molecular types as opposed to molecular instances. The many technological and social networks observed recently to be scale-free are all instance-based. Only one single airport, one single Web site, one single computer server or one single sexual partner corresponds to the associated node in the respective network. In contrast, some of the few biochemical networks being plotted and discussed in the literature thanks to the existence of experimental data (such as chemical reaction network or proteome map) are type-based.

[1] The BA model for building scale-free graphs is made of two main steps: (i) at each time step a new node with m links is added to the network (*growth*) ; (ii) the probability p_i that a new vertex will be connected to a node i is $p_i = k_i / \sum ki$, k_i being the degree of node i (*preferential attachment*). The more partners a node has the more likely this same node will be the partner of a new node that enters the network. The application of this BA law during the growing of the network, instead of a pure random attachment law (where a new node would randomly connect with existing nodes), will give more chance to some nodes to acquire a larger connectivity, driving the distribution to a power-law decay for the number of nodes as a function of their number of partners (with an exponent -3) rather than an exponential decay produced by a random growing [14].

The next sections will introduce the basic ingredients of the successive computer simulations and discuss expected results obtained by three experiments: a first one with no natural hub and the two successive ones with high frequency and low frequency hubs. The purpose of these experiments is the study the implications of these features on the modelling of complex (growing) networks. In this way, we hope to arrive at a biochemical plausible model of growing networks that can explain the observed degree distributions in biochemical data.

2 Basic Ingredients of the Computer Models

Let' s describe the basic ingredients of our tentatively more biology-like computer simulations of network structural evolution. Every node of the network is a binary string of length N so that only 2^N nodes are possible when the network has been entirely filled. Reflecting the key-lock aspect of biological or chemical binding, a node n_i will connect with another one on the basis of their hamming distance (DH). So the simplest binding rule will be for a node n_i to connect to another one n_j if:

$$DH(n_i, n_j) > t \tag{1}$$

meaning an Hamming distance (DH) superior to a given threshold t.

Eventually, each node should potentially be able to connect with other nodes. Each node i has a certain concentration that, in a very first attempt, simply changes as an effect of the on-going recruitment of nodes. When a node, randomly chosen, is recruited into the network, either it exists already and thus its concentration is just incremented by 1 or it does not exist so far and is included in the network with a concentration initially fixed to 1.

As discussed in the introduction, one *instance node* is not the same thing as one *node type* and we are interested here in growing *type-based* networks. While one instance node will be able to connect with one possible partner and only one at the moment of its recruitment, various nodes of this same type (namely the same binary string) will be able to connect to different node types. Therefore the final connectivity topology will be drawn as a function of the types of node and not of the instance nodes for which we assume one and only one partner is allowed.

The simulation goes as follows:

a) In the beginning a small number of nodes with initial concentration equal to one are recruited in the system just to kick off the growing.

b) Afterwards, at each time step, a new instance node, generated randomly, will enter the network only if it connects with an existing node type. To test this, whenever a new random instance node is proposed, a given number of trials is done with existing nodes selected probabilistically as a function of their concentration, the more concentrated the more likely to be tested. So the probabilistic preferentially of attachment is here a function of the concentration of the current nodes instead of their connectivity.

c) The test is based on the defined affinity criterion (here it is the simplest one described in Equation (1)). If following this given number of trials, the test was never successful, a new random node is proposed.
d) Once the test is succeeded and if not already existing in the network, the new node gets in and creates a new node type with initial concentration one.
e) If the incoming node n_i, although able to connect to another node n_j, already exists as a node type in the network, it is not included as a new type in the network but instead its concentration is increased by one. This explains how the concentration of the types of node composing the network can change in time, reproducing a very elementary form of dynamics.
f) If the node n_j already contains the node n_i among its partners nothing changes besides the increase in concentration of n_i. While if n_i was not yet among the partners of n_j, it will be added as such. So, on account of the *type-based* nature of our network, the computer model allows at each simulation step either to add one node type to be connected with an existing one or just one new edge between two types of node already there.
g) The simulation stops after the recruitment of an arbitrary number of types of node. The graphical results of the simulation, in the form of the now classical log-log degree distribution, is presented below (Figure 1) for N=13, t=9 and the recruitment of 4000 nodes (approximately half the number of potential types).

The resulting network's degree distribution follows an exponential decay, which is not surprising as such distribution is theoretically obtained through a random growth rule [14]. In fact, no node is favoured during the attachment of any new node since the concentration of all nodes in the network approximately remains the same. The average degree is 2.72 very much below the 378 potential nodes any node could connect with. More problematic is the very weak value of the average clustering coefficient (coefficient discussed in [3]), 0.00067, showing that two partners of a same node have very low probability to bind (as a matter of fact, most of the nodes has zero value for their local clustering coefficient). Although in strong contrast with very high values observed for instance in neural and metabolic networks, this small number essentially reflects the way the attractiveness between two nodes is defined, by means of their hamming distance. This attractiveness test makes very improbable for a same lock to have two very different keys which could bind together. In the following, the presence of natural hubs will tend to recover from this unwelcome and challenging result.

3 Similar Simulations But Allowing the Presence of Natural Hubs

One key observation done recently has been that the networks observed do not show the kind of homogeneous distribution that a random growing network would produce. These observations show a bigger heterogeneity in the connectivity of the nodes as compared to the randomly growing case. Moreover certain

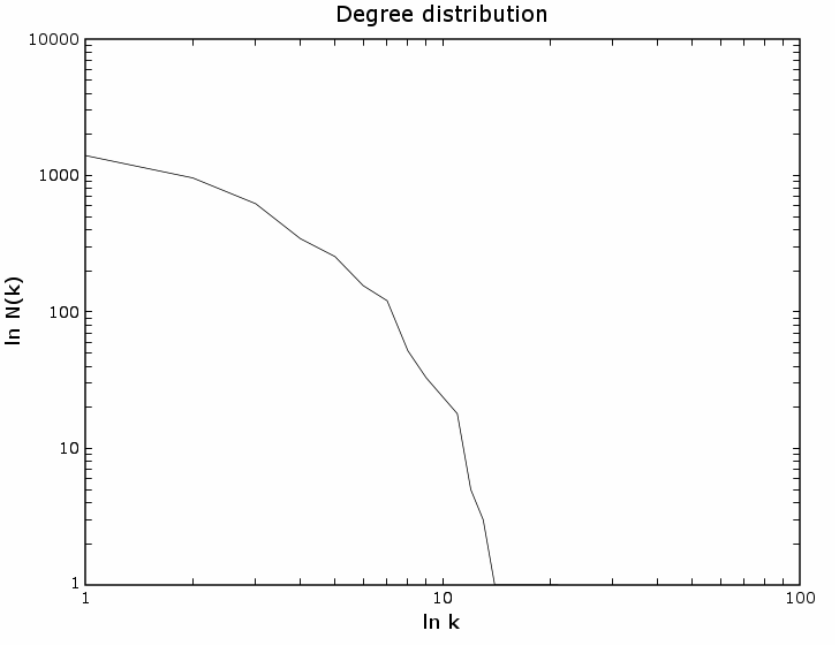


Fig. 1. The log-log degree distribution for a simulation based on the binding rule given in Equation (1). 4000 nodes are recruited in the network. The number of edges is 5457 giving an average degree of 2.72. The variance of the degree is 4.23. The clustering coefficient is 0.00067 and nearly all nodes have local clustering coefficient equal to 0. The plot takes an exponential shape typical of a random growing.

have a huge connectivity an are referred to as hubs. For most of the authors, the presence of this heterogeneity is just an outcome of the preferential attachment rule. In biology instead, types of nodes are far from identical, and some, just by their internal shaping or constitution, are more naturally inclined to play the role of hub than others. We believe this to be the main reason for the presence of hubs detected in the real biological networks; hubs are not products but simple data of history.

We propose two refinements of the computer simulations introduced in Section 2. In the first one, the frequency of hubs will not be different than the frequency of any other node types while in the second one hubs will turn out to be much rarer (as a type not in necessarily in frequency). In the two following sections these refinements will be discussed.

3.1 Refinement 1: Introducing Attractiveness

In this simulation, see Figure 2, any node is a binary string of length N and possesses an additional attribute called its *attractiveness threshold* (AT_i) comprised in between [1,N-1]. The new rule of binding is that two nodes can bind if

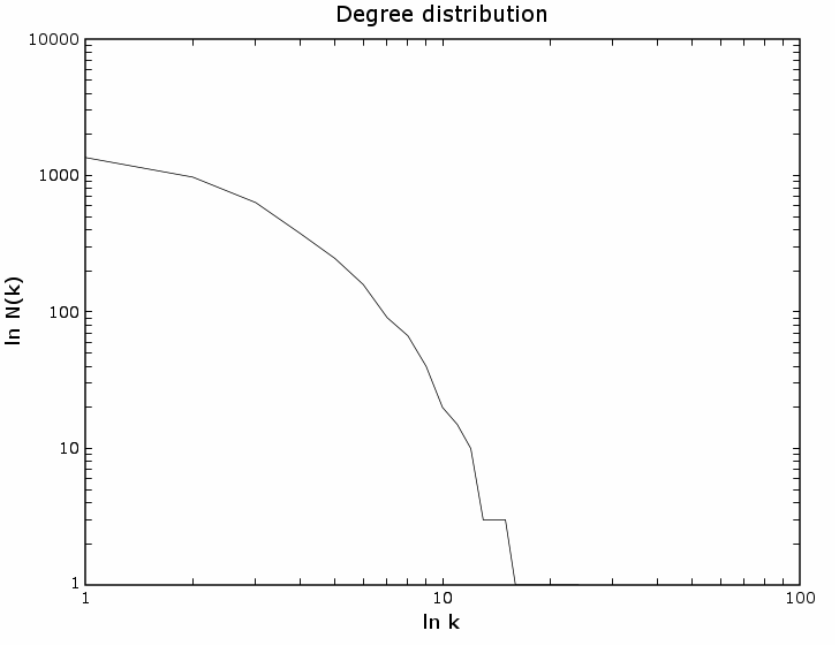


Fig. 2. The log-log degree distribution for a simulation based on the definition of nodes with *attractiveness threshold* between [0,9], node length = 10, the binding rule given in Equation (2) and high frequency hubs. 4000 nodes are recruited in the network. The number of edges is 5542 giving an average degree of 2.77. The variance of the degree is 4.51. The clustering coefficient is 0.00083. The plot takes an exponential shape typical of a random growing. The effect of natural hubs is negligible.

and only if:

$$DH(n_i, n_j) > min(AT_i, AT_j) \tag{2}$$

As a consequence, a node with a low attractiveness threshold has much more possibilities to connect than a node characterised by the same binary string but with a higher threshold. We performed a simulation with $N = 10$ [2]. A node with a small attractiveness threshold is a potential natural hub but, by the way it is defined, any type of hub is as likely as any other type. For instance, there will be as many strong hubs of length 10 (for which $AT = 1$) as any other type of length 10 and many nodes (i.e. 2^N) share a same attractiveness. Again, as shown in Figure 2, plotting the degree distribution following the recruitment of 4000 random nodes, nothing really exciting is to be pointed out since the presence of these natural hubs does not really modify the distribution. Either hub or not, all types will tend to have a same concentration and so an equal

[2] The reason for this reduction in length being to have a same potential number of types as in the previous simulation ($2^{10} * 9$) despite the additional attribute differentiating types with same bit strings.

chance to be selected. The curve is slightly shifted on the left with respect to the previous simulation (average degree = 2.77) and the presence of natural hubs slightly increases the average clustering coefficient to 0.00083. This is a situation that the last kind and most innovative simulation proposed in this paper will challenge. In this new set up, we want the hubs to be naturally much less frequent than the other types of node. The idea is that there are less biological ways to become a natural hub than an anonymous node. For a node to present one set of characteristics which makes it appealing for a certain category of potential nodes it is something, but possessing all characteristics to make it appealing for all categories of partners is a complete different story. On the whole, hubs should be harder and thus less frequent to appear in any environment, biological or technological.

3.2 Refinement 2: Inverting Dependence Between Hubness and Type Frequency

To achieve an inverse dependency between the *hubness* and the frequency, a node type can now have its length varying between 1 and N bits (so that at the end, the potential number of cell is equal to $2^{N+1} - 2$, we take $N = 12$ for the simulation). Nodes are defined so that the smaller one type is in size the more potential connections it can present. The new rule of binding is defined as follows. Let's call l_i the length of node n_i and make the new distance DHL between two nodes i and j of different length to be the Hamming distance between the l_i bits of n_i (here we will suppose l_i smaller than l_j) and l_i contiguous bits of n_j beginning from a position selected randomly between 1 and $(N - l_i)$. The two nodes will bind if:

$$DHL(n_i, n_j) = Min(l_i, l_j) \tag{3}$$

To some extent, the random choice of the position of the beginning bit to compare aims at reproducing the spatial orientation the two biological entities must adopt in order to bind or interact. Also, by avoiding a node to only connect to a complementary node with contiguous opposite bits departing always from the same constant position, the probability of forming triangular clusters is increased, an apparently important facet of biological networks. A node of length 1 (here *0* and *1*) becomes a very strong hub, able to connect to nearly every other nodes, while nodes of length N are just able to connect to a very limited amount of nodes. An interesting consequence of this new binding rule is that *a natural power law emerges simply by the definition of the nodes and the way they can connect.* It is indeed elementary to verify that for a given length l, each one of the 2^l nodes can approximately connect to $2^{(N-l+1)} + (l - 2)$ other nodes so that the function relating the number of nodes with their potential number of partners adopts a power-law shape. In other words, we get *scale-free for free* and the complete network with all types and all edges is inherently scale-free, independently of any growing mechanism. Results of the simulation (the log-log degree distribution) for

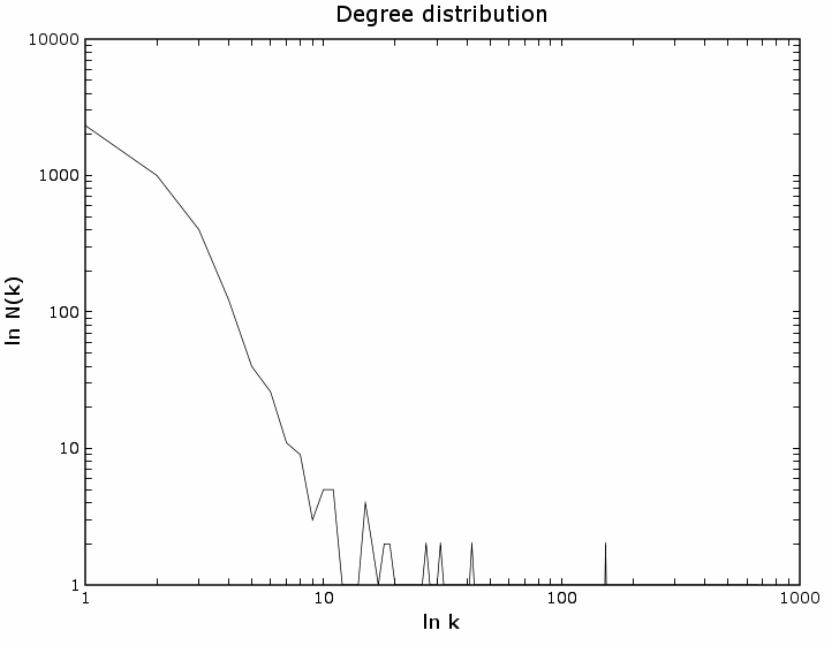


Fig. 3. The degree distribution for a simulation based on the definition of nodes with varying length between 1 and 12, the binding rule given in Equation (3), and in the presence of low frequency hubs, the smaller in length the stronger in attractiveness. 4000 nodes are recruited in the network. The number of edges is 5619 giving an average degree of 2.80. The variance of the degree is 353.07. The clustering coefficient is 0.040 with many nodes having a local clustering coefficient above 0.5. The plot takes an exponential shape typical of a random growing but now the effect of natural hubs is largely visible in the distribution tail, showing numerous picks at very high degree.

$N = 12$ and again the random generation of 4000 nodes (all nodes whatever their length have equal chance to be generated) is plotted in Figure 3. Once again this distribution is shaped as an exponential decay coming out of the randomness of the recruitment, of the attachment rules and the somewhat homogeneity of the concentration, but a key difference appears in the tail of the curve: some peaks come into view, testifying for the presence of the natural hubs. Despite the maintenance of an exponential decay for small degrees, the effect of natural hubs, although less represented than poorly connected nodes, is very patent for higher degrees and makes the first, the second and the third simulations as well as their respective topologies very distinct in the final part of the distribution. Despite an average degree which is still small (2.81), strong hubs show up. For instance, the ten top hubs with their respective degree are: *1* (670), *0* (601), *11* (463), *100* (272),*00* (263), *01* (233), *110* (205), *10* (153), *000* (153), degrees much bigger than the average value. An exponential decay for low degree seems to be still compatible with the presence of hubs in the network at much higher degree (a very heavy and rugged tail), a

presence that just mirrors their existence in reality, independently of any history of growing.

4 Discussion

Is preferential attachment a realistic method for growing biological networks? Based on biological observation, corresponding networks are more likely to grow in a less constrained and more random way, possibly producing the type of exponential distribution given by the various simulations presented in this paper. The question then becomes how these simulations correspond to the observations made in biological data. Since biological data is obtained using a relatively small amount of nodes, one can wonder whether the observed power-law distributions is correct. Besides hubs, high clustering and all the biological functional properties they could be responsible for are still compatible with an exponential decay at low degree, although their presence is often and (perhaps wrongly) written to be conditioned by a scale-free topology. When noticing the large influence of the concentration increase in the topology obtained by the simulations (mainly the third one), we need to restrain from definitive claims on network topology until at least a deeper attention is paid to the interplay between the concentration dynamics and the meta-dynamics, the original project of many authors adept of Alife and studying the evolution of biological networks 15 years ago [17,18,19,20].

A remarkable observation coming from our simulations with the varying nodes length is that in spite of a natural topology inherently scale-free, the simulation of the growing network somewhat counters this intrinsic topology by *insisting* in producing again an exponential distribution. This again illustrates the importance of the sampling done through the biological species in order to see how they do connect. A certain sampling could produce an exponential topology while another one, performed on the same biological system but selecting different species, would produce a scale-free version of it. This dependency on the sampling should be taken into account more carefully in, for instance, the study of protein-protein interaction map composition, which is undertaken by a lot of researchers, yet with very different outcomes [21,22]. Moreover, a static sampling could produce a different topology than the one spontaneously adopted by the network when preserved in its natural environmental conditions.

References

1. Barabási,A.-L.; Albert, R. : Emergence of scaling in random networks. Science **286**, (1999). pp. 509-512.
2. Ferrer i Cancho, C., Janssen R., Solé, R.V. : The topology of technology graphs: small world pattern in electronic circuits. Phys. Rev.(2001). E **63**, 32767.
3. Newman, M.E.J. : The structure and function of complex networks. SIAM Review. **45** (2003). pp. 167-256
4. Pastor-Satorras, R., Vespignani, A. : Evolution and structure of the Internet: A statistical physics approach. Cambridge University Press (2004)

5. Dorogovtsev, D., Mendes, J.F.F. : Evolution of networks: From biological nets to the Internet and WWW. Oxford University Press (2003).
6. Solé, R.V., Pastor-Satorras, R., Smith, E., Kepler, T. : A model of large-scale proteome evolution. Adv. Complex Syst. **5**, (2002) pp. 43-54.
7. Strogatz, S. : Exploring Complex Networks. Nature **410** (2001) pp. 268-276.
8. Barabási, L-A., Oltvai, Z. N. : Network Biology: Understanding the cell's functional organization. Nature Reviews Genetics. **5**. (2004). pp. 101-113.
9. Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., Barabási, A-L. : The large-scale organisation of metabolic networks. Nature **407**, (2000) 651–654.
10. Uetz, P. et al. : A comprehensive analysis of protein-protein interactions in Saccharomes cerevisiaie Nature **403** (2000) - pp. 623–627
11. Vazquez, A., Flamimi, A., Maritan, A. and Vespignani, A. : Modeling of Protein Interaction Networks In ComplexUs **1**, (2003) pp. 38–44.
12. Wagner, A., Fell, D.A. : The small world inside large metabolic networks. Proc. R. Soc. Lond. B. **268**. (2001). pp. 1803–1810.
13. Wagner, A. : How the global structure of protein interaction networks evolve. Proc. R. Soc. London B **270**. (2003) pp. 457–466.
14. Barabsi, A.-L., Albert, R., Jeong, H. : Mean-field theory for scale-free random networks Physica A 272, 173-187 (1999).
15. Temkin, O.N. and Zeigarnik, A.V. and Bonchev D. : Chemical reaction networks: a graph-theoretical approach, CRC Press (1996).
16. Vogelstein B., Lane D., Levine, A. J. : Surfing the p53 network Nature **408**, 307–310 (2000).
17. Bersini, H. : Immune Network and Adaptive Control. In Toward a Practice of Autonomous Systems. Proceedings of the first European Conference on Artificial Life, Varela and Bourgine, (1993) 217-225. MIT Press.
18. De Boer, R.J., Perelson, A.S. : Size and Connectivity as Emergent Properties of a Developing Immune Network Journal of Theor. Biology, **149** (1991) pp. 381–424
19. Detours, V., Bersini, H., Stewart, J., Varela, F.: Development of an Idiotypic Network in Shape Space, Journal of Theor. Biol. **170**, (1994)). pp. 401–404.
20. Varela, F., Coutinho, A. : Second Generation Immune Network, Immunology Today, **12** No 5 (1991) pp. 159-166.
21. Thomas, A., Cannings, R., Monk, N.A.M., Cannings, C. : On the structure of protein interaction networks Biochem. Soc. Trans. **31**, (1491–1496) (2003)
22. Michael P. H. Stumpf, Carsten Wiuf, Robert M. May Subnets of scale-free networks are not scale-free: Sampling properties of networks PNAS **102** 12 4221–4224 (2005)

Multi-population Cooperative Particle Swarm Optimization

Ben Niu[1,2], Yunlong Zhu[1], and Xiaoxian He[1,2]

[1] Shenyang Institute of Automation, Chinese Academy of Sciences,
Shenyang, 110016, China
[2] School of Graduate, Chinese Academy of Sciences,
Beijing, 100039, China
{niuben, ylzhu}@sia.cn

Abstract. Inspired by the phenomenon of symbiosis in natural ecosystem, a master-slave mode is incorporated into Particle Swarm Optimization (PSO), and a Multi-population Cooperative Optimization (MCPSO) is thus presented. In MCPSO, the population consists of one master swarm and several slave swarms. The slave swarms execute PSO (or its variants) independently to maintain the diversity of particles, while the master swarm enhances its particles based on its own knowledge and also the knowledge of the particles in the slave swarms. In the simulation part, several benchmark functions are performed, and the performance of the proposed algorithm is compared to the standard PSO (SPSO) to demonstrate its efficiency.

1 Introduction

The particle swarm optimization (PSO) algorithm, first developed by Kennedy and Eberhart [1, 2], has already come to be widely used in many areas [3, 4]. However, it was pointed out that although PSO can show significant performance in the initial iterations, the algorithm might encounter problems in reaching optimum solutions efficiently for several approximation problems [5]. This indicates that particle swarm lost its diversity and all the particles were attracted towards the best position so far by any of particles. Research addressing the shortcomings of PSO is ongoing and includes such changes as dynamic or exotic sociometries [6, 7, 8], spatially extended particles that bounce [9], increased particle diversity [10, 11], evolutionary selection mechanisms [12], and of course tunable parameters in the velocity update equations [13, 14].

The foundation of PSO is based on the hypothesis that social sharing of information among conspecifics offers an evolutionary advantage [1]. It reflects the cooperative relationship among the individuals (fish, bird, insect) within a group (school, flock, swarm). However, in natural ecosystem, many species have developed cooperative interactions with other species to improve their survival. Such cooperative—also called symbiosis—co-evolution can be found in organisms going from cells (e.g., eukaryotic organisms resulted probably from the mutualistic interaction between prokaryotes and some cells they infected) to superior animals

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 874 – 883, 2005.

(e.g., African tick birds obtain a steady food supply by cleaning parasites from the skin of giraffes, zebras, and other animals), including the common mutualism between plants and animals (e.g., pollination and seed dispersion in change of food) [15, 16].

Inspired by this research, a multi-population cooperation particle swarm optimization (MCPSO) is proposed in this paper, which is devoted to solve the problems of premature convergence and lacking in diversity encountered in many applications of PSO.

The paper is organized as follows: Review of SPSO is provided in section 2. Description of the proposed algorithm MCPSO is given in section 3. Next, experimental settings and experimental results are given in section 4. Finally, section 5 concludes the paper.

2 Standard Particle Swarm Optimization (SPSO)

The basic PSO is a population based optimization tool, where the system is initialized with a population of random solutions and the algorithm searches for optima by updating generations. In PSO, the potential solutions, called particles, fly in a *D*-dimension search space with a velocity which is dynamically adjusted according to its own experience and that of its neighbors.

The *i*th particle is represented as $\vec{x}_i = (x_{i1}, x_{i2}, ..., x_{iD})$, where $x_{id} \in [l_d, u_d]$, $d \in [1, D]$, l_d, u_d are the lower and upper bounds for the *d*th dimension, respectively. The rate of velocity for particle *i* is represented as $\vec{v}_i = (v_{i1}, v_{i2}, ..., v_{iD})$, is clamped to a maximum velocity vector $\vec{v}_{\max}$, which is specified by the user. The best previous position of the *i*th particle is recorded and represented as $P_i = (P_{i1}, P_{i2}, ..., P_{iD})$, which is also called $pbest$. The index of the best particle among all the particles in the population is represented by the symbol g, and p_g is called $gbest$. At each iteration step, the particles are manipulated according to the following equations:

$$v_{id} = wv_{id} + R_1 c_1 (P_{id} - x_{id}) + R_2 c_2 (p_{gd} - x_{id}) \tag{1}$$

$$x_{id} = x_{id} + v_{id}, \tag{2}$$

where w is inertia weight [16]; c_1 and c_2 are acceleration constants; and R_1, R_2 are random vectors with components uniformly distributed in [0, 1]. For Eq. (1), the portion of the adjustment to the velocity influenced by the individual's own *pbest* position is considered as the cognition component, and the portion influenced by *gbest* is the social component. After the velocity is updated, the new position of the *i*th particle in its dth dimension is recomputed. This process is repeated for each dimension of the *i*th particle and for all the particles in the swarm.

3 Multi-population Cooperative Particle Swarm Optimization

In MCPSO, firstly $N \times n(N \geq 2, n \geq 2)$ particles are initialized, and the particles can be divided into N swarms (one master swarm and N-1 salve swarms). Each slave swarm with n particles adapts itself according to its own evolutionary strategy independently. For the master swarm, the particles enhance themselves based on not only the social knowledge of the master swarm but also that of the slave swarms. This idea was realized by further incorporating a new dimension on the velocity of the particles in standard PSO. The resulting equations for the manipulation of the master swarm are:

$$v^{M}_{id} = w\, v^{M}_{id} + R_1 c_1 (p^{M}_{id} - x^{M}_{id}) + R_2 c_2 (p^{M}_{gd} - x^{M}_{id}) + R_3 c_3 (p^{S}_{gd} - x^{M}_{id}), \tag{3}$$

$$x^{M}_{id} = x^{M}_{id} + v^{M}_{id}, \tag{4}$$

where M represents the master swarm, c_3 is the migration coefficient, and R_3 is a uniform random sequence in the range [0, 1]. Note that the particle's velocity update in the master swarm is associated with three factors:

i. p^{M}_{id} : Previous best position of the master swarm.

ii. p^{M}_{gd} : Best global position of the master swarm.

iii. p^{S}_{gd} : Previous best position of the slave swarms.

As Shown in Eq. (3), the first term of the summation represents the inertia (the particle keeps moving in the direction it had previously moved), the second term represents memory (the particle is attracted to the best point in its trajectory), the third term represents cooperation (the particle is attracted to the best point found by all particles of master swarm) and the last represents information exchange (the particle is attracted to the best point found by the slave swarms).

The initialized populations are distributed with different regions of the search space of the entire feasible solution to maintain the diversity of the individuals. In this paper, the following population scheme is proposed (see Fig. 1): a master-slave communication model is used to assign fitness evaluations and maintain algorithm synchronization. Independent populations are associated with nodes, called slave swarms. Each node independently executes PSO or its variants, including the update of particles' position and velocity, and the creation of a new local population. When all nodes are ready with the new generations, each node then sends the best local individual（particle）to the master node. The master node selects the best one of all received individuals and evolves according to Eqs. (3) and (4). The pseudo-code for the MCPSO algorithm is listed in Fig. 2.

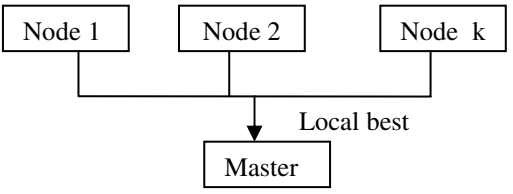


Fig. 1. The master-slave model

Algorithm MCPSO
Begin
Initialize all the populations
Evaluate the fitness value of each particle
Repeat
Do in parallel
Node I, 1≤ i ≤ K //K is the number of slaver swarms
End Do in parallel
Barrier synchronization //wait for all processes to finish
Select the fittest local individual p_g^S from the slave swarms
Evolve the mast swarm
//Update the velocity and position using Eqs. (3) and (4), respectively
Evaluate the fitness value of each particle
Until a terminate-condition is met
End

Fig. 2. Pseudo-code for the MCPSO algorithm

4 Experiment and Result

In this section, three nonlinear benchmark functions that are commonly used in evolutionary computation literature [13, 17] are performed. All functions are designed to have minima at the origin.

1. Rosenbrock function

$$f_1(x)=\sum_{i=1}^{n}100\times(x_{i+1}-x_i^2)^2+(1-x_i)^2 \quad \text{Global minimum: } x_i=1, f_1(x)=0 \tag{5}$$

2. Rastrigrin function

$$f_2(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10 \quad \text{Global minimum: } x_i=0, f_2(x)=0 \tag{6}$$

3. Griewank function

$$f_3(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1 \quad \text{Global minimum: } x_i = 0, f_3(x) = 0 \tag{7}$$

To evaluate the performance of the proposed MCPSO, two variants of SPSO are used for comparisons: SPSO1 [18], SPSO2 [18]. The parameters used for SPSO1 are recommended from Shi and Eberhart [18] with an asymmetric initialization method and a linearly decreasing w which change from 0.9 to 0.4. SPSO2 provides a transitional comparison to SPSO1 as a symmetric initialization. In our case, three populations of SPSO1 are involved in MCPSO as slave swarms to optimize the above benchmark functions and each of them has the same parameter settings as SPSO1. For SPSO1, SPSO2 and MCPSO, $x_{\max}$ and $v_{\max}$ are set to be equal and their values for f_1, f_2 and f_3 are 100, 10, 600, respectively. The acceleration constants c_1 and c_2 for SPSO1and SPSO2 are both 2.0. The acceleration constants $c_1 = c_2 = 2.05$ and migration coefficient $c_3 = 0.8$ are used in MCPSO. The parameters setting for all algorithms are summarized in Table 1.

Table 1. Parameter setting

Type	SPSO1	SPSO2	MCPSO
initialization	asymmetric	symmetric	asymmetric
Inertia weight	0.9 to 0.4	0.9 to 0.4	0.9 to 0.6
number of swarms	1	1	4
c_1	2.0	2.0	2.05
c_2	2.0	2.0	2.05
c_3	—	—	0.8

Table 2. Mean fitness values for Rosenbrock function

P	*Dim*	*G*	SPSO1	SPSO2	MCPSO
20	10	1000	96.1715	44.1374	4.5229
	20	1500	214.6764	87.2810	19.6717
	30	2000	316.4468	132.5973	17.5155
40	10	1000	70.2139	24.3512	4.1054
	20	1500	180.9671	47.7243	11.1590
	30	2000	299.7061	66.6341	15.7867
80	10	1000	36.2945	15.3883	3.1391
	20	1500	87.2802	40.6403	10.2597
	30	2000	205.5596	63.4453	33.4982
160	10	1000	24.4477	11.6283	2.6322
	20	1500	72.8190	28.9142	17.3233
	30	2000	131.5866	56.6689	18.8168

Table 3. Mean fitness values for Rastrigrin function

P	*Dim*	*G*	SPSO1	SPSO2	MCPSO
20	10	1000	5.5572	5.2062	3.5848
	20	1500	22.8892	22.7724	12.9345
	30	2000	47.2941	49.2942	37.8082
40	10	1000	3.5623	3.5697	2.5869
	20	1500	16.3504	17.2975	11.9404
	30	2000	38.5280	38.9142	28.2703
80	10	1000	2.5379	2.3835	0.9950
	20	1500	13.4263	12.9020	8.9546
	30	2000	29.3063	30.0375	23.879
160	10	1000	1.4943	1.4418	0.9948
	20	1500	10.3696	10.0438	5.9698
	30	2000	24.0864	24.5105	12.934

Table 4. Mean fitness values for Griewank function

P	*Dim*	*G*	SPSO1	SPSO2	MCPSO
20	10	1000	0.0919	0.0920	0.0503
	20	1500	0.0303	0.0317	0.0068
	30	2000	0.0182	0.0482	0.0035
40	10	1000	0.0862	0.0762	0.0602
	20	1500	0.0286	0.0227	0.0078
	30	2000	0.0127	0.0153	0.0031
80	10	1000	0.0760	0.0658	0.0462
	20	1500	0.0288	0.0222	0.0106
	30	2000	0.0128	0.0121	0.0007
160	10	1000	0.0628	0.0577	0.0406
	20	1500	0.0300	0.0215	0.0098
	30	2000	0.0127	0.0121	0.0022

In order to investigate whether MCPSO scales well or not, different population sizes P are used for each function with different dimensions. The population sizes P are 20, 40, 80 and 160. The maximum number of generations G is set as 1000, 1500, and 2000 corresponding to the dimensions 10, 20 and 30, respectively. For fair comparison, in all cases the population size of SPSO1 and SPSO2 is the same as those used in MCPSO (i.e., for each population size used in MCPSO, P/N particles are allocated for each sub-swarm). A total of 50 runs for each experimental setting are conducted.

Table 2 to Table 4 list the mean fitness values of the best particle found for the 50 runs for the three benchmark functions. From the Tables, MCPSO outperforms SPSO1 and SPSO2 significantly for almost all the cases. In a general analysis of all

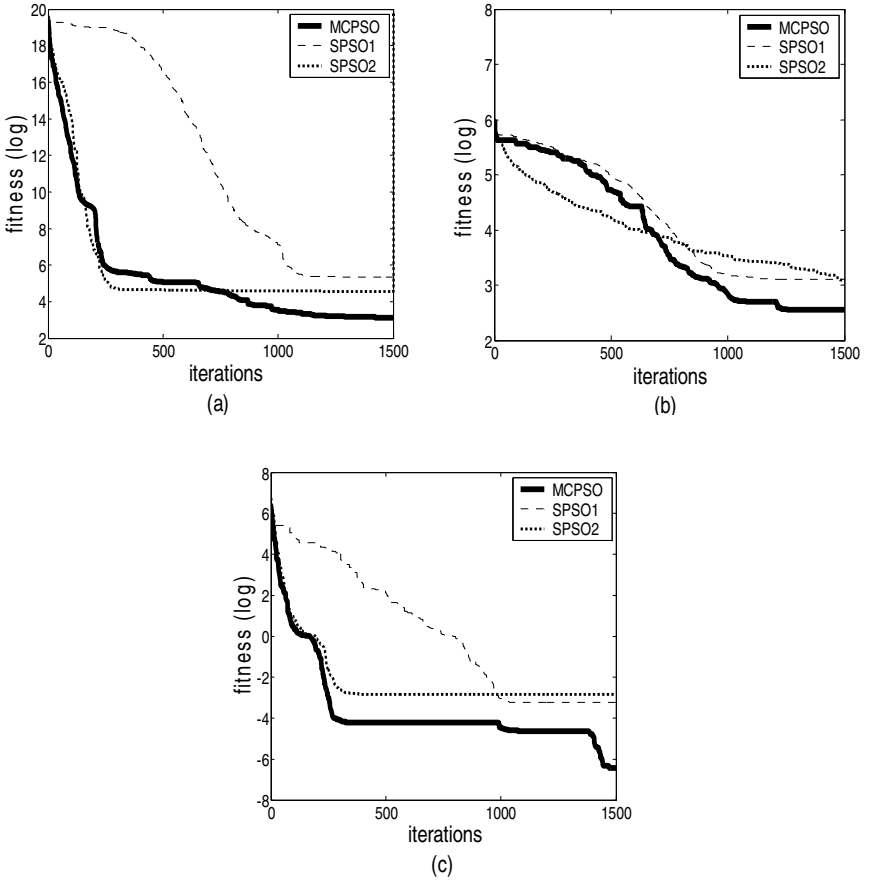


Fig. 3. Evolution of average fitness for MCPSO and SPSO when optimizing: a) f_1, Rosenbrock function, b) f_2, Rastrigrin function, c) f_3, Griewank function (notice that the scale is natural logarithmic)

tables we may conclude that MCPSO with different population sizes has almost the similar performance and scales well for all three functions. Fig. 3 (a) to Fig. 3 (c) show the mean fitness value of the best particles found during 1500 generations with 20 dimensions by 20 particles for f_1, f_2 and f_3 respectively. With the same setting with linearly decreasing w, SPSO2 is superior to SPSO1 for f_1, and is similar to SPSO1 for f_2 and f_3. The little difference of the results between SPSO1 and SPSO2 verifies that PSO is only slightly affected by the asymmetric initialization. However, for all the functions the result generated by MCPSO is better than those generated by SPSO1 and SPSO2. The graphs presented in Fig. 3 (a) to Fig. 3 (c) illustrate MCPSO will sustainable evolve when SPSO1 and SPSO2 is almost stagnated.

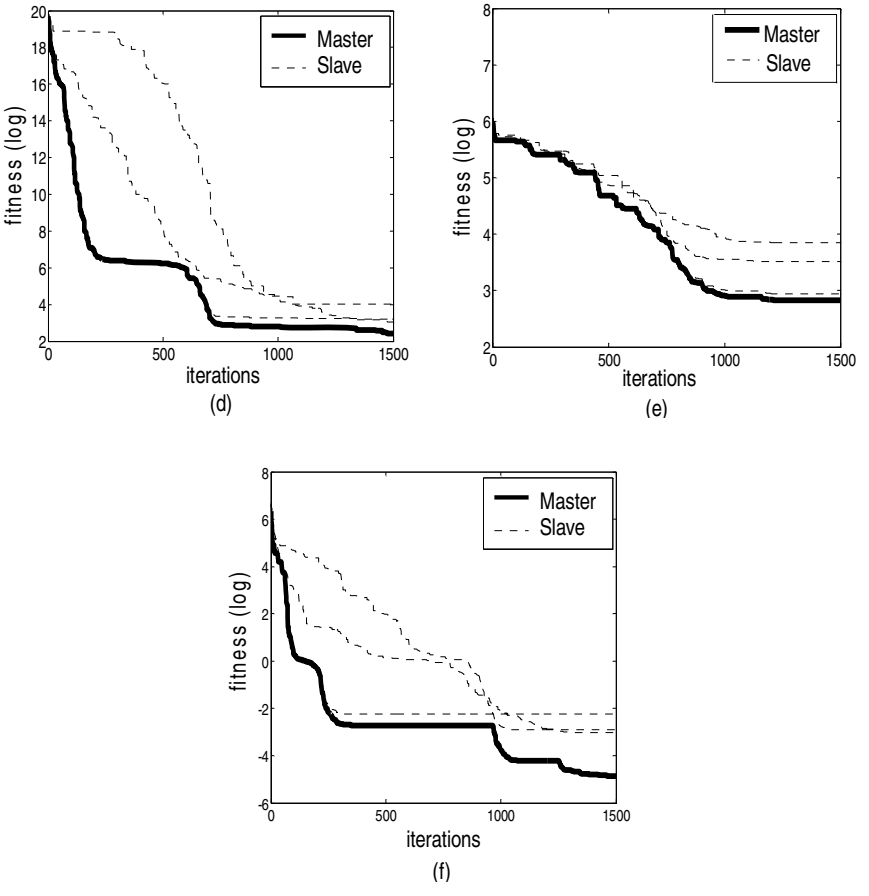


Fig. 4. Multi-population evolutionary process for benchmark functions: d) f_1, Rosenbrock function, e) f_2, Rastrigrin for function , f) f_3, Grienwank function (notice that the scale is natural logarithmic)

Fig. 4 (d) to Fig. 4 (f) show the multi-population evolutionary process for the three benchmark functions, where the population sizes P is set as 20, and the maximum number of generations G is set as 1500 corresponding to the dimensions 20. By looking at the shapes of the curves in the graphs, it is easy to see that each particle in the master swarm can keep track of the previous best position found by slave swarms, as well as find a better position based on its own knowledge. In fact, since the competition relationships of the slave swarms, the master swarm will not be influenced much when a certain slave swarms get stuck at a local optima. It may be concluded that the results generated by MCPSO is more robust than SPSO1 and SPSO2.

5 Conclusions and Future Work

This paper presents a biologically inspired mechanisms designed to improve the performance of SPSO. A symbiosis mechanism is implemented in addition to the SPSO, which result in a new algorithm (called MCPSO) characterized by a master-slaver mode. In MCPSO, one master swarm lives in symbiosis with several slave swarms. The evolution of slave swarms is likely to amplify the diversity of individuals of populations and consequently to generate more promising particles for the master swarm. The master swarm updates the particle states based on both its own experience and that of the slave swarms.

This new method is capable of controlling the balance between exploration and exploitation. Three benchmark functions are performed in the simulation part using different algorithms. The performance comparisons indicate that MCPSO is superior to SPSO in both the high quality of the solution and the robustness of the results.

Future work is focused on optimizing the performance of the proposed MCPSO. In addition, extensive application on more complicated practical optimization tasks is necessary to fully investigate the properties and evaluate the performance of MCPSO.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 70431003) and the National Basic Research Program of China (Grant No. 2002CB312200). The first author would like to thank Prof. Q.H Wu of Liverpool University for reviewing the manuscript.

References

1. Eberchart, R. C., Kennedy, J.: A new optimizer using particle swarm theory. In: proceeding of the 6th international symposium on Micromachine and Human Science, Nagoya, Japan (1995) 39-43
2. Kennedy, J., Eberhart, R. C.: Particle Swarm Optimization. In: proceeding. of IEEE International Conference on Neural Networks, Piscataway, NJ (1995) 1942-1948
3. Eberchart, R.C., Shi, Y.: Particle swarm optimization: developments, applications and resources. In: Proceedings of the IEEE Congress on Evolutionary Computation, Piscataway, NJ (2001) 81-86
4. Kennedy, J., Eberhart, R. C., Shi, Y.: Swarm intelligence. Morgan Kaufmann Publishers, San Francisco (2001)
5. Angeline, P.J.: Evolutionary optimization versus particle swarm optimization: philosophy and performance difference. In: Proceeding of the 7th Annual Conference on Evolutionary Programming, San Diego,. California USA (1998) 601-610
6. Richards, M., Ventura, D.: Dynamic sociometry in particle swarm optimization. In: International Conference on Computational Intelligence and Natural Computing, Cary, North Carolina USA (2003) 1557-1560
7. Kennedy, J., Mendes, R.: Population structure and particle swarm performance. In: Proceedings of the Congress on Evolutionary Computation, Honolulu, Hawaii (2002) 1671-1676

8. Kennedy, J., Mendes, R.: Neighborhood topologies in fully-informed and best-of neighborhood particle swarms. In: Proceedings of the 2003 IEEE SMC Workshop on Soft Computing in Industrial Applications, Binghamton, New York (2003) 45-50
9. Krink, T., Vestertroem, J.S., Riget, J.: Particle swarm optimization with spatial particle extension. In: Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii (2002) 1474-1479
10. Riget, J., Vesterstroem, J.S.: A diversity-guided particle swarm optimizer – the ARPSO. Technical Report 2002-02, Department of Computer Science, University of Aarhus (2002)
11. Løvbjerg, M.: Improving particle swarm optimization by hybridization of stochastic search heuristics and self-organized criticality. Master's thesis, Department of Computer Science, University of Aarhus (2002).
12. Angeline, P. J.: Using selection to improve particle swarm optimization. In: Proceedings of the 1998 IEEE Congress on Evolutionary Computation, Piscataway, NJ (1998) 84-89
13. Shi, Y., Eberhart, R. C.: Parameter selection in particle swarm optimization. In: proceeding of the 7th annual conference on Evolutionary Programming, San Diego, California USA (1998) 591-600
14. Clerc, M., Kennedy, J.: The particle swarm: Explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6 (2002) 58 - 73
15. Moriartv, D., Miikkulainen: Reinforcement learning through symbiotic evolution. Machine learning, 22 (1996) 11-32
16. Wiegand, R. P.: An analysis of cooperative co-evolutionary Algorithms. PhD thesis, George Mason University, Fairfax, Virginia (2004)
17. Shi, Y., Ebrehart, R. C.: A modified particle swarm optimizer. In: Proceeding of the1998. IEEE International Conference on Evolutionary Computation, Piscataway, NJ (1998) 69-73
18. Shi, Y., Eberhart, R. C.: Empirical study of particle swarm optimization. In: proceedings of the 1999 IEEE Congress on Evolutionary Computation, Piscataway NJ (1999) 1945-1950

On Convergence of Dynamic Cluster Formation in Multi-agent Networks

Mikhail Prokopenko[1], Piraveenan Mahendra Rajah[2], and Peter Wang[1]

[1] CSIRO Information and Communication Technology Centre,
Locked bag 17, North Ryde, NSW 1670, Australia
[2] 2/177, Jeffcott Street, North Adelaide, SA 5006, Australia

Abstract. Efficient hierarchical architectures for reconfigurable and adaptive multi-agent networks require dynamic cluster formation among the set of nodes (agents). In the absence of centralised controllers, this process can be described as self-organisation of dynamic hierarchies, with multiple cluster-heads emerging as a result of inter-agent communications. Decentralised clustering algorithms deployed in multi-agent networks are hard to evaluate precisely for the reason of the diminished predictability brought about by self-organisation. In particular, it is hard to predict when the cluster formation will converge to a stable configuration. This paper proposes and experimentally evaluates a predictor for the convergence time of cluster formation, based on a regularity of the inter-agent communication space as the underlying parameter. The results indicate that the generalised "correlation entropy" K_2 (a lower bound of Kolmogorov-Sinai entropy) of the volume of the inter-agent communications can be correlated with the time of cluster formation, and can be used as its predictor.

1 Introduction

Dynamic creation and maintenance of "optimal" hierarchies in large dynamic networks is a well-recognised challenge. It appears in many different contexts, e.g., as dynamic hierarchies in Artificial Life [15], coalition formation in Agent-based Systems [17], decentralised clustering in Multi-Agent Systems [12], dynamic cluster formation in Mobile Ad Hoc Networks [10], etc. In this paper, we consider a sub-problem from this class: dynamic cluster formation in a sensor and communication network without centralised controllers. This process can be described as *self-organisation* of dynamic hierarchies, with multiple cluster-heads emerging as a result of inter-agent communications. Importantly, the emphasis is on rules of interactions (or communication protocols) between the engaged lower-level entities (cells, agents, network nodes, etc.) and the structures and patterns emerging at a higher-level (multi-cellular boundaries, multi-agent coalitions, local hierarchs or cluster-heads, etc.).

In general, the clustering of sensor-data aims at grouping entities with similar characteristics together so that main trends or unusual patterns may be discovered. Self-organising cluster formation in multi-agent networks/systems has two specific primary challenges: a) decentralised clustering: even if a correct classification can be determined with the incomplete information available, the location of items belonging to a class also needs to be discovered, "data is widely distributed, data sets are volatile, or data items cannot be compactly represented" [12], and; b) dynamic (on-line) clustering: new

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 884–894, 2005.

events may require reconfiguration of clusters: the resulting patterns or clusters have to be constantly refined. This requires efficient algorithms for decentralised sensor-data clustering in a distributed multi-agent system. A method for grouping networked agents with similar objectives or data without collecting them into a centralised database is presented by Ogston et al. [12], and shows very good scalability and speed in comparison with the k-means clustering algorithm. It employs a heuristic for breaking large clusters when required, and a sophisticated technique dynamically matching agents objectives, represented as connections in the multi-agent network.

However, decentralised clustering algorithms deployed in multi-agent networks are hard to evaluate precisely for the reason of the diminished predictability brought about by self-organisation. In particular, it is hard to predict when the cluster formation will converge to a stable configuration. Such a predictive ability is, however, important for deciding whether clusters will form in time for multi-agent diagnostics, being a prerequisite for the overall damage propagation prognosis. The specific objective of this paper is an identification and evaluation of potential predictors for the convergence time of dynamic cluster formation.

In achieving this goal, we analyse two levels of multi-agent dynamics: macro-level, where coordination patterns form and can be observed, and micro-level, where the inter-agent messages are exchanged, creating a multi-agent communication space. We consider irregularity of the inter-agent communication space, and propose it as a possible predictor for our task. This predictor is estimated via the generalised "correlation entropy" K_2 of the underlying time series: the traffic volume of inter-agent communications. The estimates are shown to be correlated with the convergence time of cluster formation.

The experiments required to evaluate the predictor were carried out on a self-monitoring sensor and communication network developed CSIRO-NASA "Ageless" Aerospace Vehicle (AAV) project, in the context of Structural Health Management (SHM). The AAV project is briefly described in the next section, followed by a simplified version of a decentralised adaptive clustering algorithm developed for evaluation purposes. Section 3 presents the proposed predictor for the convergence time of cluster formation, followed by a discussion of the obtained results and future work.

2 Adaptive Clustering in Self-organising SHM Networks

Structural health monitoring and management of complex, safety-critical structures such as aerospace vehicles will ultimately require the development of intelligent networks systems that can process the data from large numbers of sensors; evaluate and diagnose detected damage; form a prognosis for the damaged structure; make decisions regarding response to or repair of the damage; initiate the required actions and monitor their effectiveness [13,3]. Recently, several essential concepts for self-organising SHM networks as well as their desirable characteristics, such as robustness, reliability and scalability, have been identified in the literature [13,14]. Some of these concepts are being developed, implemented and tested in the AAV Concept Demonstrator (AAV-CD): a hardware multi-cellular sensing and communication network whose aim is to detect and react to impacts by projectiles that, for a vehicle in space, might be micro-meteoroids or space debris. A stand-alone Asynchronous Simulator capable of simulating the AAV-

CD dealing with some environmental effects such as particle impacts of various energies has been developed and used in the reported experiments. The damage sensing network may consists of "cells" (agents) that not only form a physical shell ("skin") for a structure (e.g., an aerospace vehicle), but also have passive sensors detecting elastic waves generated in the "skin" by impacts; and electronic modules, acquiring data from the sensors, running the agent software and controlling the communications with its neighbouring cells. Importantly, a cell should communicate only with immediate neighbours, eliminating single critical points of failure: all data are processed locally, and only information relevant to other regions of the structure is communicated.

Single cells may detect impacts and triangulate their locations, while collections of cells may solve more complex tasks. Some responses could be purely local, while some may require emergence of dynamic reconfigurable structures, with some cells taking the roles of "local hierarchs". A cluster-head may be dynamically selected among the set of nodes and become a local coordinator of transmissions within the cluster. A typical SHM task may require impact-data clusters, logically grouping the cells which detected impacts with energies within a certain band (e.g., non-critical impacts). Moreover, clusters would form and re-form when new damage is detected on the basis of local sensor signals. Importantly, a cluster formation algorithm should be robust to changes caused by new impacts, cells' failures and possible repairs.

As pointed out earlier, our main goal is not a new clustering method *per se*, but rather an analysis of a representative clustering technique in a dynamic and decentralised multi-agent setting, exemplified by the AAV sensor and communication network, *in terms of predictability of its convergence time*. There are some important specific details of our experimental setup which may be relevant to other multi-agent networks: a particular communication infrastructure where each cell is connected only to immediate neighbours; constraints on the communication bandwidth; dynamic scenarios where density of events may vary in time and space; a decentralised architecture without absolute coordinates or id's of individual cells on a large-scale multi-cellular skin. To stay within a generic framework, we abstracted away almost all sensor-data features. For example, instead of considering time-domain or frequency-domain impact data, detected and/or processed by cell sensors [13], we represent a cell sensory reading with a single aggregated value ("impact-energy"), define "differences" between cells in terms of this value, and attempt to cluster cells while minimising these "differences". This approach can be relatively easily extended to cases where "differences" are defined in a multi-dimensional space. In short, our focus is on inter-agent communications required by a decentralised clustering algorithm, dynamically adapting to changes, and the convergence time.

2.1 Dynamic Cluster Formation Algorithm

The algorithm input can be described as a series (a flux) of events (impacts) detected at different times and locations, while the output is a set of non-overlapping clusters, each with a dedicated cluster-head (a network cell) and a cluster map of its followers (cells which detected the impacts) in terms of their sensor-data and relative coordinates. The algorithm is described elsewhere [11] and involves a number of inter-agent messages notifying agents about their sensory data, and changes in their relationships and actions. For example, an agent may send a recruit message to another agent, delegate the role

of cluster-head to another agent, or declare "independence" by initiating a new cluster. Most of these and similar decisions are based on the clustering heuristic described by Ogston et al. [12], and a dynamic offset range. This heuristic determines if a cluster should be split in two, and the location of this split.

Each cluster-head (initially, each agent) broadcasts its *recruit* message periodically, with a broadcasting-period, affecting all agents with values within a particular dynamic offset ε of the impact-energy data x detected by this agent. Every *recruit* message contains the sensor-data of all current followers of the cluster-head with their relative coordinates (a cluster map). Under certain conditions, an agent, which is not a follower in any cluster, receiving a *recruit* message becomes a follower, stops broadcasting its own *recruit* messages and sends its information to its new cluster-head indicating its relative coordinates and the sensor-data x. However, there are situations when the receiving agent is already a follower in some cluster and cannot accept a recruit message by itself — a recruit disagreement. In this case, this agent *forwards* the received recruiting request to its present cluster-head. Every cluster-head waits for a certain period, collecting all such *forward* messages, at the end of which the clustering heuristic is invoked on the union set of present followers and all agents who *forwarded* their new requests.

Firstly, all n agents in the combined list are sorted in decreasing order according to their impact-energy value x. Then, a series of all possible divisions in the ordered set of agents is generated. That is, the first ordering is a cluster with all agents in it; the second ordering has the agent with the largest value in the first cluster and all other agents in the second cluster; and so forth (the n-th division has only the last n-th agent in the second cluster). For each of these divisions, the quality of clustering is measured by the total square error:

$$E_j^2 = \sum_{i=1}^{z} \sum_{x \in A_{i,j}} \|x - m_{i,j}\|^2 \ ,$$

where z is a number of considered clusters ($z = 2$ when only one split is considered), $A_{i,j}$ are the clusters resulting from a particular division and $m_{i,j}$ is the mean value of the cluster $A_{i,j}$. We divide E^2 values by their maximum to get a series of normalised values. Then we approximate the second derivative of the normalised errors per division:

$$f''(E_j^2) = (E_{j+1}^2 + E_{j-1}^2 - 2E_j^2) \ / \ h^2 \ ,$$

where $h = \frac{1}{n}$. If the peak of the second derivative is greater than some threshold for a division j, we split the set accordingly; otherwise, the set will remain as one cluster.

The cluster-head which invoked the heuristic notifies new cluster-heads about their appointment, and sends their cluster maps to them: a *cluster-information* message. When the clustering heuristic is applied, it may produce either one or two clusters as a result. If there are two clusters, the offset of each new cluster-head is modified. It is adjusted in such a way that the cluster-head of the "smaller" agents (henceforth, references like "larger" or "smaller" are relative to the value x) can now reach up to, but not including, the "smallest" agent in the cluster of "larger" agents. Similarly, the cluster-head of "larger" agents can now reach down to, but not including, the "largest" agent (the cluster-head) of the cluster of "smaller" agents. These adjusted offsets are sent to the new cluster-heads along with their cluster maps.

There are other auxiliary messages involved in the algorithm but importantly, the cluster formation is driven by three types: *recruit*, *cluster-information*, and *forward* messages. The first two types are periodic, while the latter type depends only on the

degree of disagreements among cluster-heads. On the one hand, if there are no disagreements in clustering (for instance, if a clustering heuristic resulted in optimal splits even with incomplete data), then there is no need in *forward* messages. On the other hand, when cluster-heads frequently disagree on formed clusters, the *forward* messages are common. In short, it is precisely the number of *forward* messages traced in time — the traffic volume of inter-agent communications — that we hope may provide an underlying time series $\{v_t\}$ for our prognostic analysis, as it exhibits both periodic and chaotic features.

The quality of clustering is measured by the weighted average cluster diameter [23], but the algorithm does not guarantee a convergence minimising this criterion. In fact, it may give different clusterings for the same set of agent values, depending on the physical locations of the impact points. The reason is a different communication flow affecting the adjustment of the offsets. Each time the clustering heuristic is executed in an agent, its offsets are either left alone or reduced. The scope of agents involved in the clustering heuristic depends on the order of message passing, which in turn depends on the physical locations of impacts. The adjusted offsets determine which agents can be reached by a cluster-head, and this will affect the result of clustering. Therefore, for any set of agent values, there are certain sequences of events which yield better clustering results than others.

We conducted extensive preliminary simulations to determine whether the algorithm is robust and scales well in terms of the quality of clustering and convergence, measured by the number of times the clustering heuristic was invoked before stability is achieved with each data set [11]. Several scenarios were considered. The first scenario kept the network size constant, while increasing the number of impacts detected within it. The second scenario, on the contrary, fixed the number of impacts, while increasing the network size. In other words, the density of impacts was increasing in the first case, and decreasing in the second. Finally, we developed scenarios where impacts appear periodically, with varying periods. While the simulation results show that the algorithm converges and scales well in all cases, and in addition, is robust to dynamics of the sensor-data flux, the convergence time varies significantly (Figure 1), without obvious indicative patterns.

In the remainder of the paper we focus on our main objective: prediction of the convergence time T, based on regularity of an initial segment $0, ..., \mathcal{D}$, where $\mathcal{D} < T$, of the "communication-volume" series $\{v(t)\}$, where $v(t)$ is the number of *forward* messages at time t.

3 The $K_q(\mathcal{D})$ Predictor: Entropy of Multi-agent Communication-Volume

The observed variability of different communication-volume time series may indicate that the underlying dynamics in the phase-space includes both unstable periodic and chaotic orbits, and an unstable fixed-point. It is known that in many experiments, time series often exhibit irregular behavior during an initial interval before finally settling into an asymptotic state which is non-chaotic [1] — in our case, eventually converging to a fixed-point ($v_T = 0$). The irregular initial part of the series may, nevertheless, contain valuable information: this is particularly true when the underlying dynamics is deterministic and exhibits *transient chaos* [1,7]. We believe that the described algorithm

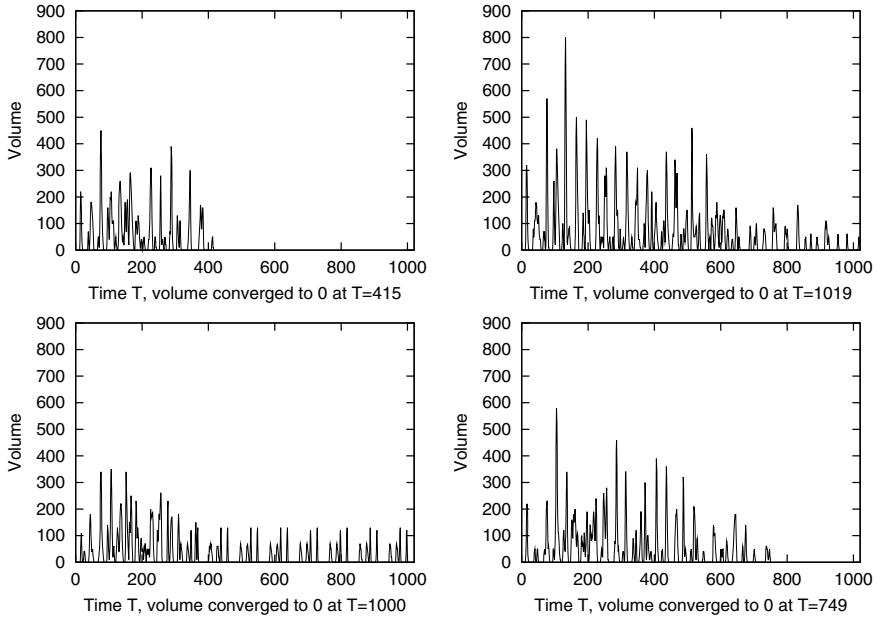


Fig. 1. Varying convergence times T_s for 4 different experiments, $1 \leq s \leq 4$

for dynamic cluster formation, employing the clustering heuristic and adjustments of the offset ε, creates *multi-agent transient chaotic dynamics*.

Our plan is simple: for each experiment s, a) select an initial segment of length $\mathcal{D}$ of the time series, $\{v_s^{\mathcal{D}}\}$; b) estimate its generalised entropy $K_q(\mathcal{D})_s$ for a range of estimation-dependent parameters (see the description below). Then, c) given the estimates $K_q(\mathcal{D})_s$ for all the experiments, correlate them with the observed convergence times T_s, e.g., by using a linear regression $T = a + bK_q(\mathcal{D})$ and the correlation coefficient ρ between the series $\{T_s\}$ and $\{K_2(\mathcal{D})_s\}$. This would allow us to predict the time T_s of convergence to $v_s(T_s) = 0$, as $T_s = a + bK_q(\mathcal{D})_s$.

A simple characterisation of the "regularity" of the communication space is provided by the auto-correlation function of an integer delay τ:

$$\gamma_s(\tau) \;=\; \sum_{t=\tau+1}^{\mathcal{D}} [v_s(t-\tau) - \overline{v_s}]\,[v_s(t) - \overline{v_s}] \;/\; \sum_{t=1}^{\mathcal{D}} [v_s(t) - \overline{v_s}]^2 \,. \qquad (1)$$

The auto-correlation is obviously limited to measuring only linear dependencies, and we consider instead a more general and elaborate approach. One classical measure is the Kolmogorov-Sinai (KS) entropy, also known as metric entropy [8,9,18]: it is a measure for the rate at which information about the state of the system is lost in the course of time. In other words, it is an entropy per unit time, or an "entropy rate". Suppose that the $d-$dimensional phase space is partitioned into boxes of size r^d. Let $P_{i_0 \ldots i_d\ _1}$ be the joint probability that a trajectory is in box i_0 at time 0, in box i_1 at time Δt, ..., and in box i_{d-1} at time $(d-1)\Delta t$, where Δt is the time interval between measurements on the state of the system (in our case, we may assume $\Delta t = 1$, and omit the limit $\Delta t \to 0$ in the following definitions). The KS entropy is defined by

$$K = -\lim_{\Delta t \to 0} \lim_{r \to 0} \lim_{d \to \infty} \frac{1}{d\Delta t} \sum_{i_0 \ldots i_d \ \ 1} P_{i_0 \ldots i_d \ \ 1} \ln P_{i_0 \ldots i_d \ \ 1} \,, \tag{2}$$

and more precisely, as a supremum of K on all possible partitions. This definition has been generalised to the order-q Rényi entropies K_q [16]:

$$K_q = -\lim_{\Delta t \to 0} \lim_{r \to 0} \lim_{d \to \infty} \frac{1}{d\Delta t(1-q)} \ln \sum_{i_0 \ldots i_d \ \ 1} P^q_{i_0 \ldots i_d \ \ 1} \,. \tag{3}$$

It is well-known that $K = 0$ in an ordered system, K is infinite in a random system, and K is a positive constant in a deterministic chaotic system. Grassberger and Procaccia [4] considered the "correlation entropy" K_2 in particular, and capitalised on the fact $K \geq K_2$ in establishing a sufficient condition for chaos $K_2 > 0$. The Grassberger and Procaccia (GP) algorithm estimates the entropy K_2 as follows:

$$K_2 = \lim_{r \to 0} \lim_{d \to \infty} \lim_{N \to \infty} \ln \frac{C_d(N,r)}{C_{d+1}(N,r)} \,, \tag{4}$$

where $C_d(r)$ is the correlation integral:

$$C_d(N,r) = \frac{1}{(N-1)N} \sum_{i=1}^{N} \sum_{j=1}^{N} \Theta(r - \|\boldsymbol{V_i} - \boldsymbol{V_j}\|) \,. \tag{5}$$

Here Θ is the Heaviside function (equal to 0 for negative argument and 1 otherwise), and the vectors $\boldsymbol{V_i}$ and $\boldsymbol{V_j}$ contain elements of the observed time series $\{v(t)\}$, "converting" or "reconstructing" the dynamical information in one-dimensional data to spatial information in the d-dimensional embedding space: $\boldsymbol{V_k} = (v_k, v_{k+1}, v_{k+2}, \ldots, v_{k+d-1})$ [19]. The norm $\|\boldsymbol{V_i} - \boldsymbol{V_j}\|$ is the distance between the vectors in the d-dimensional space, e.g., the maximum norm [20]:

$$\|\boldsymbol{V_i} - \boldsymbol{V_j}\| = \max_{\tau=0}^{d-1} (v_{i+\tau} - v_{j+\tau}) \tag{6}$$

Put simply, $C_d(r)$ computes the fraction of pairs of vectors in the d-dimensional embedding space that are separated by a distance less than or equal to r. In order to eliminate auto-correlation effects, the vectors in Equation (5) should be chosen to satisfy $|i - j| > L$, for some positive L, and at the very least $i \neq j$ [21]. Since we consider only an initial segment of the times series, we simply set $N = \mathcal{D}$ in the Equation (5), estimating the entropy as

$$K_2(d, r, \mathcal{D}) = \ln \frac{C_d(\mathcal{D}, r)}{C_{d+1}(\mathcal{D}, r)} \,. \tag{7}$$

Now we only need to identify the embedding dimension $\hat{d}$ and the distance $\hat{r}$ which maximise the correlation coefficient for s experiments, $\rho(\{T_s\}, \{K_2(d, r, \mathcal{D})_s\})$, over a range of d and r, and designate

$$K_2(\mathcal{D})_s = K_2(\hat{d}, \hat{r}, \mathcal{D})_s. \tag{8}$$

At this stage we need to make a comment on the correlation dimension. Within certain ranges of r and d, the correlation integral $C_d(r)$ may be proportional to some power of

r, $C_d(r) \sim r^\nu$ [5]. This power ν is called the correlation dimension. If the dynamical process is unfolded by choosing a sufficiently large $d > d_\nu$, a typical slope of the plot $\ln C_d(r)$ versus $\ln r$ becomes independent of d. We observed (see the next section) that this minimum embedding dimension d_ν is not necessarily the embedding dimension $\hat{d}$ maximising the predictor $K_2(\mathcal{D})$, but a possible connection is intriguing.

The correlation dimension provides useful information about the local structure of the process and is an effective measure of its (possibly fractal) size: in particular, a random process has an "infinite" correlation dimension (its orbit is not expected to have any spatial structure). In contrast, the correlation dimension for a periodic orbit is 1, while it could be higher for some non-regular processes. A non-integer $\nu < 1$ is an indication of a strange chaotic attractor [5]. It is worth pointing out that the GP algorithm can be used to estimate the correlation dimension of underlying chaotic transients [1].

4 Experimental Results

The experiments included $s = 1, ..., 20$ runs of the clustering algorithm, tracing the communi-cation-volume time series $\{v_t\}$. As expected, the auto-correlation function $\gamma_s(\tau)$, Equation (1), did not advance us in our experiments: the highest correlation coefficient between convergence times T_s and auto-correlations $\gamma_s(\tau)$, for a range of delays τ, was only 0.52.

We then selected an initial segment $\mathcal{D} = 400$, and computed correlation integrals $C_d(400, r)$ for a wide range of embedding dimensions ($d < 98$) and distances ($1 \leq r \leq 1000$, a median standard deviation of $\{v\}_s$ being about 100). A plot $\ln C_d(r)$ versus $\ln r$ is shown in Figure 2, illustrating the time series depicted in the top-left of Figure 1 (a quickly converged series, $T = 415$). We can observe three well-known regions: 1) the lower region distorted by fluctuations due to the small number of points, 2) a linear "scaling" region where the power law $C_d(r) \sim r^\nu$ holds, and 3) the upper region distorted due to the finite size of the process. There are also anomalous shoulders in the correlation integral due to remaining autocorrelation in the time-series data [22]. We observed that quickly converged series have an earlier onset of the upper region than slowly converged series. The plot strongly indicates a transient multi-mode process, and suggests the possibility of extracting meaningful predictors $K_2(\mathcal{D})_s$, specified by Equations (7)-(8).

Given data of s experiments: the 3-dimensional array $K_2(d, r, \mathcal{D})_s$ for varying d and r, and each s, the correlation coefficient $\rho(\{T_s\}, \{K_2(d, r, \mathcal{D})_s\})$ was determined for the range of d and r. It is shown in Figure 3, clearly reaching maximum at embedding dimensions $6 \leq d \leq 10$, almost uniformly for all the distances r. The maximum ($\rho = 0.898345$) was attained at $\hat{d} = 6$ and $\hat{r} = 41$, and is a very encouraging correlation value. Figure 4 shows the linear regression between $\{T_s\}$ and $\{K_2(\mathcal{D})_s\}$, where the latter is selected according to the Equation (8) for the identified $\hat{d}$ and $\hat{r}$.

5 Conclusions and Future Work

We considered decentralised and dynamic cluster formation in multi-agent sensor and communication networks, proposed and experimentally evaluated a predictor for the convergence time of cluster formation. The predictor $K_2(\mathcal{D})$ is based on the generalised

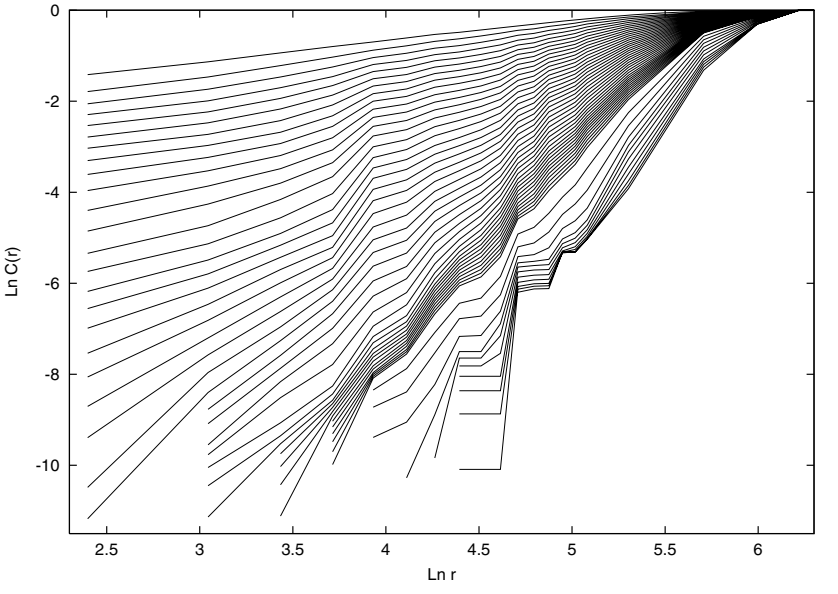


Fig. 2. A plot $\ln C_d(r)$ versus $\ln r$ for a quickly converged series, $T = 415$. Embedding dimensions are shown for every d between 1 (from top-left corner) and 40, and for every 5th d between 45 and 95 (towards bottom-right corner).

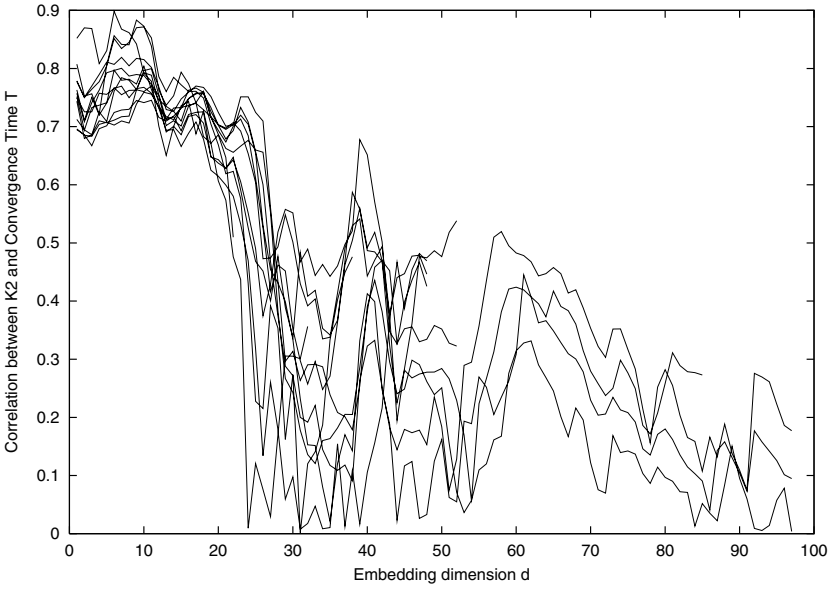


Fig. 3. The correlation coefficient ρ between the series $\{T_s\}$ and $\{K_2(\mathcal{D})_s\}$, for a range of distances r. The maximum ($\rho = 0.898345$) is at $\hat{d} = 6$ and $\hat{r} = 41$.

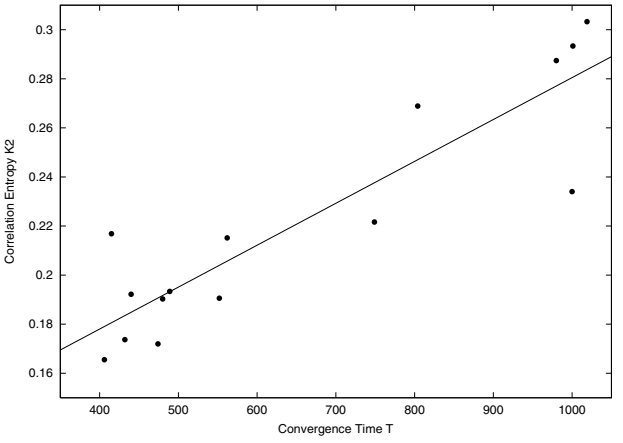


Fig. 4. The linear regression between $\{T_s\}$ and $\{K_2(\mathcal{D})_s\}$

"correlation entropy" (a lower bound of Kolmogorov-Sinai entropy) of the volume of the inter-agent communications. The results indicate that $K_2(\mathcal{D})$ can be well correlated with the time of cluster formation. The predictor $K_q(\mathcal{D})$ can also be considered for other orders q, and this work is ongoing.

The dynamic cluster formation may be interpreted in self-referential terms: inter-agent messages contribute to emergence of cluster hierarchies at macro-level, and at the same time are significantly influenced by them at micro-level. Such an interdependency can also be characterised in terms of tangled hierarchies exhibiting Strange Loops [6]: "an interaction between levels in which the top level reaches back down towards the bottom level and influences it, while at the same time being itself determined by the bottom level". The observed *multi-agent transient chaotic dynamics* may appear precisely due to this self-referentiality.

The performance of the predictor $K_2(\mathcal{D})$ provides a very good support for deploying other, more sophisticated algorithms in the sensing networks. The density-based algorithms may particularly be relevant in our application: e.g., DBSCAN algorithm would allow us to discover clusters with arbitrary shape [2].

Acknowledgements. The authors are grateful to Ed Generazio and William Prosser (NASA Langley Research Center), Daniel Polani (University of Hertfordshire), Frank Horowitz (CSIRO Exploration and Mining), and to other members of the AAV team, especially Don Price and Tony Farmer (CSIRO Industrial Physics), for their encouragement and insightful comments.

References

1. Dhamala, M., Lai, Y.C., Kostelich, E.J. Analyses of transient chaotic time series. *Physical Review E*, 64, 056207, 1–9, 2001.
2. Ester, M., Kriegel, H., Sander, J., and Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. The 2nd International Conference on Knowledge Discovery and Data Mining, 226-231, 1996.

3. Foreman, M., Prokopenko, M., Wang, P. Phase Transitions in Self-organising Sensor Networks. Banzhaf, W., Christaller, T., Dittrich, P., Kim, J.T. Ziegler, J. (Eds.) *Advances in Artificial Life - Proceedings of the 7th European Conference on Artificial Life*, 781–791, LNAI 2801, Springer, 2003.
4. Grassberger, P. and Procaccia, I. Estimation of the Kolmogorov entropy from a chaotic signal. *Physical Review A*, 28(4):2591, 1983.
5. Grassberger, P. and Procaccia, I. Characterization of strange attractors. *Physical Review Letters*, 50:346–349, 1983.
6. Hofstadter, D.R. *Godel, Escher, Bach: An Eternal Golden Braid*. New York: Vintage Books, 1989.
7. Jánosi, I.M., and Tél, T. Time series analysis of transient chaos. *Physical Review E*, 49(4):2756–2763, 1994.
8. Kolmogorov, A.N. A new metric invariant of transient dynamical systems and automorphisms in Lebesgue spaces. *Doklady Akademii Nauk SSSR*, 119:861-864 (Russian), 1958.
9. Kolmogorov, A.N. Entropy per unit time as a metric invariant of automorphisms. *Doklady Akademii Nauk SSSR*, 124:754–755 (Russian), 1959.
10. Lin R., and Gerla, M. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 1265–1275, September 1997.
11. Mahendra rajah, P., Prokopenko, M., Wang, P., Price, D.C. Towards Adaptive Clustering in Self-monitoring Multi-Agent Networks. The 9th International Conference on Knowledge Based and Intelligent Information and Engineering Systems (KES-2005), Melbourne, Australia, September 2005.
12. Ogston E., Overeinder, B., Van Steen, M., and Brazier, F. A Method for Decentralized Clustering in Large Multi-Agent Systems. The 2nd International Joint Conference on Autonomous Agent and Multi Agent Systems, 798–796, 2003.
13. Price, D.C., Scott, D.A., Edwards, G., Batten, A., Farmer, A.J., Hedley, M., Johnson, M., Lewis, C., Poulton, G., Prokopenko, M., Valencia, P., Wang, P. An Integrated Health Monitoring System for an Ageless Aerospace Vehicle. 4th Int'l Workshop on Structural Health Monitoring, Stanford, 2003.
14. Prokopenko, M., Wang, P.., Price, D.C., Valencia, P., Foreman, M., Farmer, A.J. Self-organising Hierarchies in Sensor and Communication Networks. To appear in *Artificial Life*, Special issue on Dynamic Hierarchies, 2005.
15. Rasmussen, S., Baas, N.A., Mayer, B., Nilsson, M., and Olesen, M.W. Ansatz for Dynamical Hierarchies. *Artificial Life*, vol. 7, 4, 2001.
16. Rényi, A. *Probability theory*. North-Holland, Amsterdam, 1970.
17. Sandholm, T. and Lesser, V. Coalition Formation among Bounded Rational Agents. The 14th International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, 662-669.
18. Sinai, Ya.G. On the concept of entropy of a dynamical system. *Doklady Akademii Nauk SSSR*, 124:768-771 (Russian), 1959.
19. Takens, F. Detecting strange attractors in turbulence. *Dynamical systems and turbulence*, LNM Vol.898, Springer, Berlin, 1981.
20. Takens, F. Invariants related to dimension and entropy. *Atas do 13* Colóquio Brasiliero do Matemática, Rio de Janeiro, 1983.
21. Theiler, J. Spurious dimension from correlation algorithms applied to limited time-series data. *Physical Review A*, 34(3):2427–2432, 1986
22. Theiler, J. Estimating fractal dimension. *J. of the Optical Society of America A*, 7(6):1055, 1990.
23. Zhang, T., Ramakrishnan, R., Livny, M. BIRCH:A New Data Clustering Algorithm and Its Applications. *Data Mining and Knowledge Discovery*, 1(2), 141-182, 1997.

On the Unit of Selection in Sexual Populations

Richard A. Watson

School of Electronics and Computer Science,
University of Southampton, SO17 1BJ. U.K.
raw@ecs.soton.ac.uk

Abstract. Evolution by natural selection is a process of variation and selection acting on replicating units. These units are often assumed to be individuals, but in a sexual population, the largest reliably-replicated unit on which selection can act is a small section of chromosome – hence, the 'selfish gene' model. However, the scale of unit at which variation by spontaneous mutation occurs is different from the scale of unit at which variation by recombination occurs. I suggest that the action of recombinative variation and mutational variation together can enable local optimization to occur at two different scales simultaneously. I adapt a recent model illustrating a benefit of sexual recombination to illustrate conditions for two scales of optimization in natural populations, and show that the operation of natural selection in this scenario cannot be understood by considering either scale alone.

1 Nucleotides, Genes, and Individuals

Although it is often convenient when providing evolutionary explanation to suppose that selection acts on individual organisms, in fact, in sexual populations the combination of alleles represented in an individual's genotype is not reliably transferred to its offspring [1][2]. In sexual populations the largest unit of genetic material that reproduces with reliable fidelity is a subsection of chromosome small enough to avoid being disrupted by crossover. The size of these units will be determined by the crossover rate[1], with higher rates defining smaller units, but for common purposes it is taken that the relevant unit is about the size of individual genes [1]. Accordingly, since the gene is the largest genetic unit that reproduces reliably, the gene is the largest unit on which natural selection can act [1] – hence, the well-known "selfish gene" framework [2]. These observations support fundamental axioms underlying the way evolution is defined, i.e. the change in frequencies of alleles in a gene pool [3], placing attention on the frequencies of individual alleles, not the frequencies of genotypes. Although the selective unit might seem unambiguous in evolutionary algorithms because evaluation is always applied to individuals and individual fitnesses determine reproduction, in fact the same issues apply if sexual recombination, or crossover, is used. Although *individuals* are selected to reproduce, the genetic material of individuals is broken-up by crossover, so it is only *fragments* of individuals whose frequencies can be affected by selection – hence, attention on "schema" in evolutionary computation theory [4].

[1] The probability of recombining adjacent loci (not the probability that crossover is applied to an individual, as sometimes meant in evolutionary computation). See *C* in Table 1.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 895 – 905, 2005.

Biologically, the genes on which selection acts may consist of thousands of nucleotides. New variants of genes (new alleles) are introduced by spontaneous point mutation affecting one or a small number of individual nucleotides within a gene. Accordingly, the scale of unit that is manipulated by mutation (individual nucleotides) is quite a different scale of unit from that which is manipulated by recombination (alleles of genes). However, since selection acts on whole alleles it is common in population genetics models to abstract away the nucleotide-level details and simply refer to the variant alleles of a gene by unique labels, e.g. *A* and *a*. Each allele label represents a different combination of maybe thousands of nucleotides, but if one allele is produced by mutation of the other, then they may differ in only a few nucleotide substitutions. This abstraction of the internal detail, hiding the level of individual nucleotides on which mutation acts, is appropriate for some purposes. Indeed, it sufficed perfectly well for all the population genetic results derived prior to the discovery of the molecular structure of DNA [5]. Historically, an allele is simply defined as a particulate unit of Mendelian inheritance (thus being intimately linked to the action of recombination), and the fact that, on a molecular basis, each consisted of thousands of nucleotides was not known. Following conventional population genetic models, evolutionary computation models [4] used in artificial life rarely make the distinction between genes and nucleotides. Individuals are generally modelled as binary strings where each bit is taken to be synonymous with the allele of a gene at a particular locus and mutation changes the bit between one allele and the other, 0 or 1, standing in for *A* and *a*. The problem is that this abstraction is not consistent with a model of a gene containing a thousand nucleotides and 4^{1000} different alleles – consider the probability of a reversion, or "back mutation", for example, or more generally, the probability of finding a particular allele by random mutation. This inconsistency can cause more than a mere terminological problem, especially in cases where mutation and recombination are applied together.

So, in population genetics and in evolutionary computation, does it matter whether genes are modelled as collections of nucleotides or simply abstracted into particulate alleles? There are many scenarios, lying within common assumptions, where it does not. But there are other scenarios that are biologically plausible where it does matter. The aim of this paper is to discuss the conceptual issues involved and examine the implications of moving outside common assumptions. I use some simulation results as an example just to illustrate some of the salient points. My claim is that existing definitions of the unit of selection and common simplifying assumptions preclude some interesting phenomena, and more specifically, that it is necessary in some cases to recognise more than one level of optimisation to understand the action of evolutionary systems. Cases (like those examined here) that step outside the usual simplifying assumptions are necessary for understanding the operation of evolution in natural populations. They are also necessary for understanding how to use and model evolution in artificial life experiments, and in addressing the complex relationship that these observations have to concepts of the unit of selection and related processes such as Shifting Balance Theory [6].

The following two sections introduce the basic ideas about why two levels of optimisation may be required to find fit genotypes in a fitness landscape, and how this might be provided. Section 4 describes a set of simulation experiments to illustrate the effect of changing the scales at which optimisation is applied. Specifically, I show

that optimisation at any one scale is insufficient to find fit genotypes, and a combination of optimisation scales is required. I illustrate two different ways in which these two scales can be provided, but the main idea is that mutation manipulates the nucleotides and recombination manipulates the alleles of genes. The quite simple idea that mutation and recombination afford local optimisation in different spaces by operating on different units motivates a rethink about simplistic notions of the unit of selection.

2 Background: Selective Units and Local Optima

I suggested above that there are evolutionary scenarios that afford optimisation on some unit of genetic material and also on combinations of that unit: specifically, nucleotides and combinations of nucleotides (i.e. alleles of genes). The consequences of this are the same for other scales of units, for example, alleles and combinations of alleles. In fact, discussion about the possibility of selection on combinations of alleles has quite a history in population genetics and we can use this to better understand the implications of having optimisation occur at the lower scales of nucleotides and combinations of nucleotides.

If selection acts on individual units (of whatever scale), then this has important consequences for what evolution by natural selection can and cannot do. If selection acts on individual units, then evolution can only respond to the net fitness effects of individual units. Although a piece of genetic material like an allele may have different fitness effects in different genetic backgrounds, the change in frequency of that unit over several generations cannot be controlled by its fitness in any particular background in a sexual population (as its presence in a given background is not reliably reproduced). Instead, Fisher argues, the change in frequency of an allele will be controlled by its average *fitness excess* [7], roughly, its average fitness effect over the backgrounds in which it occurs. A consequence of selection on the average effect of individual units is that if a particularly fit *combination* of units involves units that are *individually* disfavoured then, even if that combination should happen to arise in some individual, it cannot take hold in the population.

A particular instance of this may occur when a population is stuck at a local fitness peak in a fitness surface. Consider a genotype *A* that is locally optimal and a second genotype *B* that is fitter. All single allele substitutions to *A* are deleterious (by the definition of being locally optimal) so *B* (necessarily) differs from *A* by several allelic substitutions. These allelic differences are thus collectively favourable but individually unfavourable to a genotype at *A*. Since sexual recombination re-assorts alleles into the different genetic backgrounds of other individuals in the population, even if such a combination of alleles were introduced to an individual in a population located at *A*, selection would act on the alleles individually and remove them from the population. Sewell Wright considered the escape from a local fitness peak to a genotype of higher fitness to be the central problem of evolution [8], and devised Shifting Balance Theory, SBT, [6] to explain how population subdivision might enable an evolving population to achieve this. However, some consider the conditions for SBT to not be widely, if at all, available in natural populations [9].

Actually, it would be easy to facilitate selection on combinations of alleles if the crossover rate could be modified. That is, if the crossover rate were very low then alleles would not assort independently and would instead be selected as a unit, and in

the limit, the entire genotype of an *asexual* individual replicates as a whole. If an asexual population was centred on *A*, and the combination of alleles required to reach *B* were introduced to an individual in the population, then selection would have no problem in promoting the resultant *B* genotype to take over in the population. But selection on combinations of alleles has its disadvantages as well. In some circumstances it can be favourable to have selection act on the average fitness effect of individual alleles rather than on combinations of alleles. For example, when a sub-optimal combination of alleles arises in a genotype, containing some favourable alleles and some disfavoured alleles, an asexual population is unable to promote the good alleles without also promoting the bad alleles. This is the basis of the classic Fisher/Muller model for the benefit of sexual recombination [7]. In short, in some circumstances it is preferable to have selection act on individual units and in others it is preferable to have selection act on combinations of units. The former cannot select for good combinations of units unless they involve only units which are also individually good, and the latter cannot select for good individual units unless they are in a collectively good combination of units.

Fisher and Wright were concerned with the action of selection on alleles and combinations of alleles, but here we are concerned (for the most part) with nucleotides and combinations of nucleotides (alleles). The same reasoning applies: in some cases selection on individual nucleotides may be beneficial, and in others selection on alleles may be beneficial. In natural populations, it does not make sense to imagine that recombination rates are so high that selection acts on individual nucleotides. But I will argue that mutation effects local optimisation in nucleotide sequence space, and that if, at the same time, recombination is manipulating whole alleles, then this can effect a two-level process of optimisation. Such a scenario is precluded in prior models by common assumptions about the epistasis model, and also by the abstraction of nucleotide combinations into particulate alleles. In the next section I clarify what I mean by local optimisation and the different ways that evolving populations can provide it.

3 Modes of Local Optimisation

Given a combinatorial search space $<u_1,u_2,u_3,\ldots u_N>$ (where each point in the space is a combination of values for each of the units u_1 to u_N) local optimisation involves the movement of a point through this space where successive points are neighbouring or nearby. For example, *hill climbing* optimisation follows trajectories formed by moving to adjacent points that are higher in fitness. An evolving population is often conceived as a hill climbing process. However, there are two qualitatively different means by which this may be provided in evolution by natural selection (one being more common in evolutionary computation and the other being more common in population genetics) and these are often not distinguished properly: I call these *local mutation* and *direct selection*. Fig.1a. illustrates local mutation as provided by, for example, an asexual population with a low rate of spontaneous point mutation on nucleotides. Here, although selection promotes entire genotypes, mutation modifies one or a small number of nucleotides (e.g. third locus) enabling only local movement in the combinatorial space of nucleotides. This maps very naturally to bit-mutation

hill climbers used in computational optimisation. Fig.1.b. illustrates local optimisation via direct selection on individual units as provided by, for example, a sexual population that exhibits free recombination between alleles (that is, alleles assort independently during sexual reproduction). In this case, an evolving population can (with some simplifying assumptions about linkage equilibrium [10] and epistasis) be usefully described as a point in allele frequency space [8]. If selection acting on individual units makes small adjustments to the frequencies of these units, then the population has performed local movement in this space. This sense of local optimisation is the normal interpretation of evolution in population genetics models (though not with all the same terminology).

Formally, local mutation and direct selection move in different spaces, genotype sequence space and allele frequency space, respectively. But note that genotype sequence space is coincident with the vertices of allele frequency space (i.e. points where frequencies are either 1 or 0). The more important difference is that the units in the natural interpretation of local mutation (in the post-molecular-genetics era) are nucleotides, whereas the units in the natural interpretation of direct selection are alleles (Mendelian units of inheritance). If the units that the two mechanisms manipulate are not the same scale then this makes a profound difference in the 'locality' of the spaces in which this optimisation occurs. Although local optimisation at any scale can become stuck on local optima at that scale, local optimisation at different scales 'sees' different fitness landscapes, different fitness gradients, and different local optima.

Unlike Fig. 1b where alleles are abstract, Fig. 1c illustrates alleles that are each composed of many nucleotides. Here if variation introduces new alleles that differ by only one nucleotide (left of Fig. 1c) then we still have local optimisation at the nucleotide scale. In contrast, if variation introduces completely new alleles (right of Fig. 1c), this effects local optimisation at the allele scale but not at the nucleotide scale. A combination of mutation on nucleotides and recombination of whole alleles has the potential to provide optimisation at two different scales simultaneously, and when the local optima of one scale are different from those of the other scale, the interaction of the two can provide optimisation that neither one scale can provide alone.

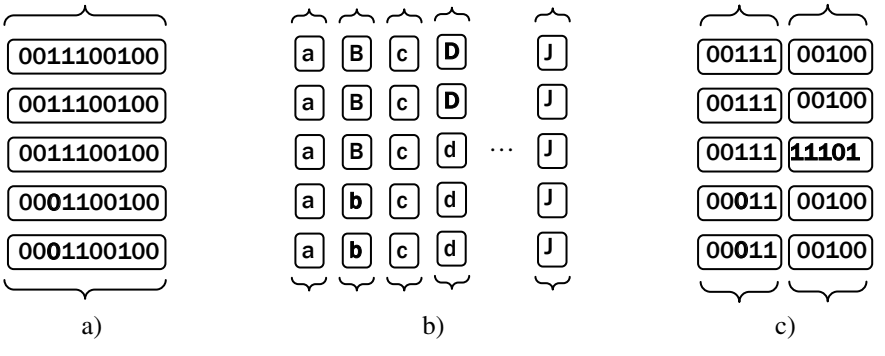


Fig. 1. Rows are individuals in a population; boxes show units of genetic material; braces show selection pools. a) local mutation. b) direct selection. c) see text.

4 Examining a Model Landscape

The consequences of optimisation at more than one scale will only be seen if we remove common simplifying assumptions about epistasis that would cause local optima at different scales to be coincident. Previous work [11] described a fitness landscape designed to illustrate a benefit of sexual recombination that is well suited for our purposes here. This work showed that a subdivided sexual population can discover the highest fitness genotypes of this landscape easily, but an asexual population (under the same conditions) cannot. In this paper I use this model to demonstrate that there are two levels of optimisation involved in this effect by explicitly examining the local mutation and direct selection mechanisms at the levels of nucleotides and alleles.

In this model a genotype consists of two genes each containing many nucleotides. The epistasis in the model has strong 'synergy' of good mutations within a gene, and also a general field of random epistatic interactions among all mutations. The fitness of a genotype is given by:[2]

$$f(G)=R_{ij}(2^{-i}+2^{-j}) \tag{1}$$

where i is the mutational (Hamming) distance of one of the genes (the first half of the genotype, $\{g_1,g_2,\ldots,g_n\}$) from an ideal allele, and j is the mutational distance from an ideal allele in the other gene (the second half of the genotype $\{g_{n+1},g_{n+2},\ldots,g_{2n}\}$), and each R_{ij} is a random value drawn uniformly in the range [0.5,1] (for one instance of this random landscape, each R_{ij} is a constant). The basic form of modularity used here, where genes are constituted by a large number of nucleotide sites that are grouped both functionally (with epistasis) and physically (by location on the chromosome), is also seen in natural systems where the nucleotides of a gene are grouped functionally and physically by virtue of the transcription and translation machinery. Without loss of generality, the maximum fitness allele for each gene can be that where all nucleotides are 1s, and the maximum fitness genotype is that where both genes have their maximum fitness alleles, i.e. the all-1s genotype. This function can be conveniently drawn as a two dimensional fitness landscape where the two axes are the number of 1s in each of the two genes (Fig. 2).

The idea behind this model is that mutation will search locally to find good combinations of nucleotides within each gene, and crossover will make new combinations of alleles to bring these the two good alleles together and hence find fit genotypes. This is a simple idea but it is worth pointing out that in most cases over-simplistic assumptions about epistasis preclude the need for a two-level description of the evolutionary process, even if the two processes are available. That is, in simple landscapes, if selection can find good alleles by finding good nucleotide mutations, then fit genotypes can be found by simply doing more of this, i.e. finding the good nucleotide mutations in all genes. However, in this landscape, it is easy to find the best allele for gene 1 by accumulating beneficial mutations only when gene 2 is not yet well optimised, and vice versa. Once either of the genes becomes well optimised it then

[2] This is modified to keep the maximum fitness value to be 3 (and make it insensitive to n, the number of nucleotides per gene) but the original shape of the function is retained.

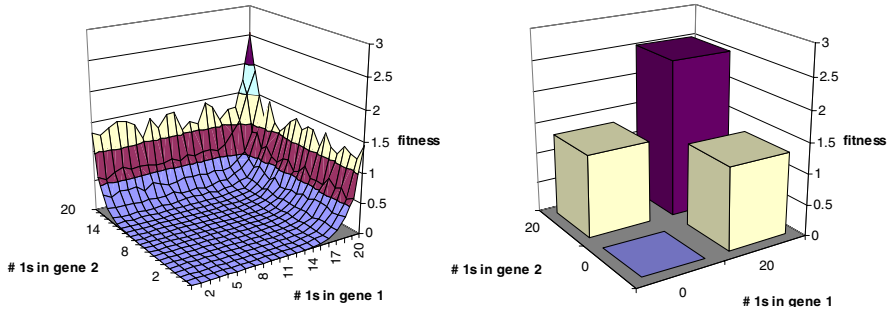


Fig. 2. The fitness landscape defined by Equation 1.left) as seen by nucleotide variation, right) as seen by combinations of good and bad alleles.

becomes difficult to find the best allele for the second gene without disrupting the fitness contributions of the first. This can be seen by considering the problem of escaping the local optima in the fitness ridges shown at the back of Fig. 2 (left). Specifically, each local optimum shown is a local optimum in nucleotide sequence space, so although it is easy to find a good combination of nucleotides for either gene by selecting on nucleotides, it is not the case that continued selection on individual nucleotides will find the fittest genotypes.

Thus local optimisation on nucleotides will easily find a good allele for one of the genes but will then be stuck on a local optimum. In a subdivided population different demes may find individuals on different local peaks, some having optimised gene-1 (but not gene-2) and others having optimised gene-2 (but not gene-1). But that is as far as it goes with optimisation only on nucleotides – to the nearest peak in nucleotide sequence space (Fig. 2. left). Additional selection on the allele scale, (taking the good alleles from different demes), can easily find fit genotypes. There are only two genes, and the best genotypes are simply the union of best alleles from these two genes (Fig. 2 right) – so local optimisation in allele space will easily find good genotypes if good alleles for these two genes are provided. However, if there were no nucleotide-scale optimisation there would be no good alleles on which this allele-scale optimisation could operate. Specifically, the number of possible combinations of nucleotides in a gene is exponential in the number of nucleotides it contains, and if fit alleles of the gene are rare then neither initial standing variation nor spontaneous mutation can be guaranteed to provide these fit alleles. In this example, the best alleles for each gene are unique in a space of 2^n possible alleles for each gene. For large n, without local optimisation in nucleotide space, finding them relies on chance and is infeasible. Local optimisation on alleles alone will only be able to select on the best alleles that are in the initial population or any better alleles that might be provided by random search in the set of alleles for each gene. Thus although optimisation at the scale of alleles is not troubled by local optima in nucleotide sequence space, it also cannot exploit local search in nucleotide space as is required to find fit alleles.

To examine the above reasoning the simulation experiments use the parameters given in Table 1. To maintain population diversity the population was subdivided with a total population of 10,000 individuals, subdivided into 100 demes of 100 individuals each. (These demes are bigger than those used in [11] so as to preclude the need to use elitism which would explicitly create a means for whole genotypes to

reproduce without modification and confound these studies). Migration between demes was such that one individual in each new generation in each deme was a migrant from some other randomly selected deme. Each sub-population independently creates a new generation by fitness proportionate selection (with replacement) [4]. Each individual in each deme is initialized to a random binary string of 100 bits – representing two genes of 50 nucleotides each. In the crossover methods, crossover is applied to all reproduction events. The data recorded are the number of generations until the first occurrence of the fittest genotype, and also the number of generations for the fittest alleles to arise in both genes (numbers in brackets give the standard deviation). 30 independent runs of each parameter set were performed. Runs were terminated at a limit of 2000 generations.

In Table 1, L=100 is the length of the genotype in bits. Mutation values are the probability that each bit of the genotype is replaced with a new random bit. 'NRA'=New Random Allele, i.e. a randomly generated combination of nucleotides for a whole gene (this is equivalent to *genewise* crossover with a random string, and thereby confirms that the success of crossover in experiment 3(b) is not the result of "macromutation" [12]). The crossover value, C, is the probability that a crossover point occurs between adjacent loci. C=0.5 is free recombination, or uniform crossover. 'Genewise' means that crossover occurs only between the two genes and never within them (given that the coding regions of biological genes are very small compared to the intergenic distances, this scenario is actually the most biologically realistic). Finally, C=1/L produces on average one crossover point per reproduction but, unlike genewise crossover, its location is random.

Table 1. Parameters and results of simulation experiments

			Generations to find:	
Experiment	**Mutation**	**Crossover**	**~ best alleles**	**~ best genotype**
1.a)	1/L	0	36 (2)	> 2000
b)	0	0.5	9 (1)	> 2000
2.a)	*NRA*	0	> 2000	> 2000
b)	0	genewise	> 2000	> 2000
3.a)	1/L	genewise	36 (2)	50 (4)
b)	0	1/L	90 (25)	181 (32)

Exp. 1 examines optimisation on the scale of nucleotides only, a) provided by local mutation on individual nucleotides (with selection on whole genotypes, i.e. C=0), b) provided by direct selection on individual alleles (i.e. C=0.5). Exp. 2 examines optimisation on the scale of whole alleles only (50 nucleotides each), a) provided by a 'local' mutation model that creates new random alleles (with selection on whole alleles via genewise crossover), b) provided by direct selection on standing variation (also via genewise crossover). Exp. 3 examines optimisation on the scales of nucleotides and alleles simultaneously, a) uses a combination of mechanisms from 1a and 2b, i.e. local mutation on individual nucleotides and direct selection on alleles. Exp. 3b uses direct selection at two scales provided by selection on variable sized sections of chromosome using a low per locus crossover probability (initial standing variation provides the necessary nucleotides).

Table 1 also gives the results of these simulations. In Exps. 1 and 2 all 30 runs failed to find the best genotype in 2000 generations. Exps. 1 and 3 succeeded in finding the best alleles for both genes in all 30 runs, but only Exp. 3 succeeds in finding the best genotypes in any runs, and in fact finds them in all runs. As hypothesised we see that optimisation on the scale of nucleotides only can find fit alleles but not fit genotypes, optimisation on the scale of alleles only cannot find fit alleles, and optimisation on both nucleotides and alleles is necessary and sufficient to find fit genotypes. Unsurprisingly, Exps. 1 and 2 are no better in other simulations using a panmictic population, failing in all 30 runs, as does Exp. 3b. in 28 of the runs. But interestingly, Exp. 3a. does succeed in 14 of 30 runs with a panmictic population (mean 71 (98)) indicating that the two-scale optimisation is not entirely dependent on population subdivision. Exp. 3b is interesting because it uses recombination to provide optimisation at both scales.[3] This appears to be dependent on the subdivision model and further examination is required to ascertain its similarities and contrasts with SBT. Comparing with the simulations performed in [11] which use a recombination rate of 1/L (as in Exp. 3b) but also use mutation at a rate of 1/L (as in 3a), Exps. 3a and 3b separate out the mechanisms that might be responsible for the result in [11].

5 Conclusions

The simulations show that, in a scenario like the example modelled, two scales of optimisation are in operation and are required to find fit genotypes. What does this tell us about the relevant units of selection? Arguably, in Exp. 3 the unit of selection is still the gene because this is the largest unit that replicates reliably under recombination at these rates. But it would be a mistake to conclude that this is the whole story. Let us consider further what has historically been the main purpose of defining the unit of selection: i.e. to identify the unit that adaptation by natural selection acts in the interest of? A lot of relevant discussion has been focussed on evolution for the good of the group versus evolution for the good of the individual [1], but also on evolution for the individual versus evolution for the gene [2]. Should we add to this list the question of evolution for the good of the gene versus evolution for the good of the nucleotide? Although these are the two scales in question here, ascribing benefit to a nucleotide does not seem conceptually useful to me. Moreover, it is not clear to me that this is the right question to be asking, or that it has a sensible answer. In contrast, I find it relatively unambiguous to state that evolution is performing local optimisation at the nucleotide scale and at the allele scale. Moreover, Exp. 3b shows that this could in principle apply at other scales such as genes and combinations of genes, and that it would be incorrect to insist that any one scale was sufficient to understand the process.

Accordingly, a preoccupation solely with the unit defined by the recombination rate does not necessarily capture all the important scales in evolutionary processes. In particular, sexual recombination and spontaneous mutation may provide different

[3] This means that multi-scale optimisation is not necessarily restricted just to nucleotides and alleles, but in principle to alleles and combinations of alleles, although the utility of optimisation at the scale of combinations of alleles would depend on the structure of the genetic map and its correspondence with epistatic interactions [13].

levels of optimisation. This suggests that it is not always appropriate to abstract the combinations of nucleotides within the alleles of genes into indivisible units simply because this is the unit that is particulate under crossover. Simplifying assumptions about epistasis may preclude the necessity for such distinctions, but biologically plausible epistatic structures are complex, and simplistic models may overlook significant structure that makes these distinctions important, as shown in the example model. Similarly, simplifying assumptions about population structure (like panmixia), and recombination models (like uniform crossover), are also self-reinforcing in that they each exclude the phenomena that make the other interesting. Natural populations often lie outside these simplifying assumptions about sex, population structure, and epistasis and require a more sophisticated treatment of the mechanisms involved.

More generally, these observations challenge our understanding of the underlying algorithmic principles of evolution by natural selection – in this example we cannot model the action of evolution as a hill-climbing process that operates at any one scale. Two-scale optimisation is more closely allied to a divide and conquer process of problem decomposition [13]. Such a distinction is implied in the evolutionary computation literature on the building block hypothesis [4][14][15]. Issues of selection on parts and wholes are also important to artificial life in understanding mechanisms that scale-up the processes of evolution [16], and to understand when evolutionary processes can do more than simple hill-climbers can [13].

References

1. Williams, G.C., 1966, *Adaptation and Natural Selection: A Critique of Some Current Evolutionary Thought.* Princeton University Press, Princeton, N.J.
2. Dawkins, R., 1976, *The Selfish Gene*, Oxford University Press, NY.
3. Hardy, G.H., 1908. Mendelian proportions in a mixed population, *Science*, **28**, 49-50.
4. Holland, J.H., 1975, *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: The University of Michigan Press.
5. Watson, J.D. & Crick, F.H., 1953, Molecular structure of Nucleic Acids, *Nature* **171**, 737-738.
6. Wright, S., 1977, *Evolution and the Genetics of Populations, Volume 3*, U. of Chicago Press.
7. Fisher, R.A., 1930. *The genetical theory of natural selection.* Dover publisher, Inc., New York.
8. Wright, S., 1931, Evolution in Mendelian populations, *Genetics* **16**: pp. 97-159.
9. Coyne, J.A., Barton, N.H., and Turelli, M., 2000, Is Wright's Shifting Balance Process Important in Evolution, *Evolution* **54**(1), pp.306-317.
10. Ridley, M. 1996. *Evolution*, 2nd ed. Cambridge, MA: Blackwell
11. Watson, R.A., 2004, A Simple Two-Module Problem to Exemplify Building-Block Assembly Under Crossover, in *Parallel Problem Solving from Nature (PPSN VIII),* Xin Yao, et al (Eds.), LNCS 3242, Springer, pp 161-171.
12. Jones, T., 1995, *Evolutionary Algorithms, Fitness Landscapes and Search*, PhD dissertation, 95-05-048, University of New Mexico, Albuquerque.
13. Watson, R.A., 2006, *Compositional Evolution: The Impact of Sex, Symbiosis, and Modularity on the Gradualist Framework of Evolution*. MIT Press, Cambridge, MA. (in press)

14. Goldberg, D.E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Reading Massachusetts, Addison-Wesley.
15. Forrest, S., & Mitchell, M., 1993a, Relative Building block fitness and the Building block Hypothesis, in *Foundations of Genetic Algorithms 2*, Whitley, D, ed., Morgan Kaufmann.
16. Maynard Smith, J.M. & Szathmary, E., 1995, *The Major Transitions in Evolution*, Freeman.

Periodic Motion Control by Modulating CPG Parameters Based on Time-Series Recognition

Toshiyuki Kondo and Koji Ito

Dept. of Computational Intelligence and Systems Science,
Interdisciplinary Graduate School of Science and Engineering,
Tokyo Institute of Technology, 4259 Nagatsuta, Midori, Yokohama 226-8502, Japan
{kon, ito}@dis.titech.ac.jp
http://www.ito.dis.titech.ac.jp/~kon/

Abstract. This paper proposes a computational motion control model of a redundant manipulator inspired by biological brain-motor systems. The proposed model consists of two processing layers dubbed "CPG" and "Dynamical memory". Likewise biological central pattern generators in spinal cord, the CPG layer plays a role in generating torque patterns for realizing periodic motions. On the contrary, the higher brain model, i.e. the Dynamical memory layer is a time-series pattern discriminator implemented by a recurrent neural networks (RNN). By associating time-series of the system states with optimized CPG parameters, the RNN can predictively modulate the generating torque patterns by recalling well-suited CPG parameters according to the sensorimotor time-series.

1 Introduction

In animal motions generation, a number of brain-nervous subsystems, such as *cerebral cortex*, *cerebellum*, *basal ganglia*, *brain stem*, *spinal cord* etc., are interdependently concerned, and the generated motions can be categorized into *voluntary movement*, *automatic movement* or *reflex* according to the concerned brain regions and the latent time. Among them, the *automatic movement*, such as walking, swimming, breathing etc. is a fundamental motion for life-sustaining, and has a periodic characteristic in common.

Walking experiments of decerebrate cats strongly evidenced the periodic movement is basically induced by spinal central pattern generator (CPG) without higher brain interference. Instead, the higher brain contributes to the generated gait pattern by activating/deactivating each CPG unit[1]. In addition, Yanagihara *et al.* reported that decerebrate cats with the cerebellum are able to learn new gait patterns under unknown environments[2]. The automatic movement is mainly realized at the lower brain (i.e. brain stem or spinal cord) based on proprioceptive feedback, while the higher brain can predictively select an appropriate motion pattern if the current situation is recognized as *known*, otherwise it starts learning how to adapt to the situation.

In our previous papers[3], we proposed a computational periodic motion control model inspired by biological brain-motor systems, and it was adopted to

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 906–915, 2005.

swing control of a simple pendulum with variable length. In this paper, we elaborate the proposed model and its applications in redundant manipulator control for crank rotation.

As shown in Fig.1, the proposed model consists of two processing layers dubbed "CPG" and "Dynamical memory". Likewise biological central pattern generator, the CPG layer plays a role in generating torque patterns for realizing periodic motions. In this study, the CPG layer is implemented by a neural oscillator model[4]. Thanks to its entrainment feature, realized periodic motions can be stabilized against environmental perturbations. This structural stabilization ability was called *global entrainment*[5]. Obviously, this stability is dependent on CPG parameters, but tuning these parameters by hand seems to be tremendously difficult since it should have lots of parameters. Thus in general, genetic algorithms (GA) or other evolutionary optimizers were utilized.

On the other hand, the Dynamical memory layer is a time-series pattern discriminator implemented by recurrent neural networks (RNN). By associating time-series observations of system states with the optimized CPG parameters, the RNN can predictively modulate the generating torque patterns by recalling well-suited CPG parameters according to the observed situation. In other words, it can recognize environmental changes (e.g. joint viscoelasticity) via its proprioceptive feedback time-series (e.g. trajectories of joint angles, angular velocities and so on).

In this paper, we describe details of the proposed model and demonstrate its validity through the simulation results of a redundant manipulator control.

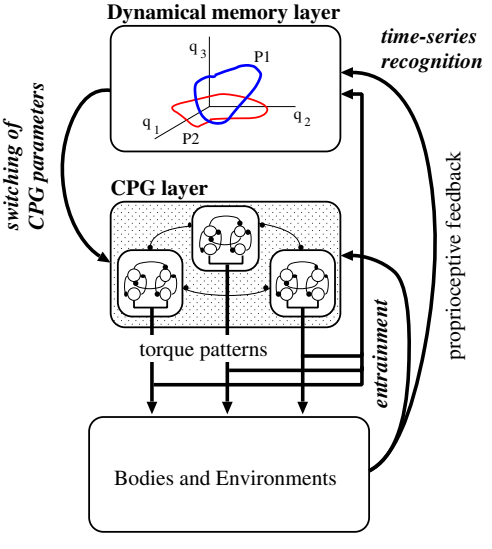


Fig. 1. A periodic motion control model

2 A Periodic Motion Generation Model

2.1 CPG Layer

As shown in Fig.1, the CPG layer has to generate a specific torque pattern for driving a controlled object suitably. In this study, we used a nonlinear neural oscillator model proposed in [4] as a torque pattern generator. As can be seen in Fig.2, dynamics of the neurons assumed in this oscillator model is represented by the following simultaneous nonlinear differential equations:

$$\tau \dot{u}_i^{[k]} = -u_i^{[k]} - w_{[k,\bar{k}]} y_i^{[\bar{k}]} - \sum_{j=1(\neq i)}^{n} w_{ij} y_j^{[k]} - \beta v_i^{[k]} + u_0 + F_i^{[k]} \tag{1}$$

$$\tau' \dot{v}_i^{[k]} = -v_i^{[k]} + y_i^{[k]}, \tag{2}$$

$$y_i^{[k]}(u_i^{[k]}) = \max(0, u_i^{[k]}), \tag{3}$$

$$\text{torque}_i = T_r \cdot \left[-y_i^{[e]} + y_i^{[f]} \right], \tag{4}$$

where y_i is a CPG output, u_i and v_i correspond to internal state variables, τ and τ' are time constants, $w_{[k,\bar{k}]}$ is a mutual inhibition weight between complementary neurons, w_{ij} is an inhibition weight among oscillators, β is a fatigue constant, u_0 is a bias input, and F_i represents a proprioceptive feedback between a controlled object and the CPG (See section 3.1). In the above equations, $[k]$ means complement representation of extensor ($[e]$) or flexor ($[f]$), i.e. if $[k]$ represents $[e]$, the complement $[\bar{k}]$ corresponds to $[f]$. The driving torque for motor

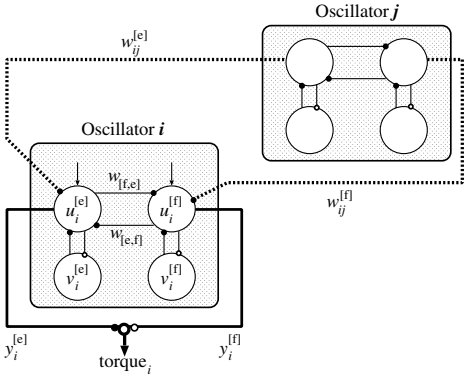


Fig. 2. CPG model

control can be generated in proportion to the difference of the outputs of the extensor and flexor neurons (equation (4)). T_r is a torque conversion coefficient.

Although Matsuoka's model has lots of parameters to be adjusted, some of the parameters, e.g. fatigue constant, can be fixed because it has less correlation with the frequency and the amplitude of the generated torque pattern[6]. Furthermore, it is known that fixing the ratio of the time constants ($\tau/\tau' = const.$) has an effect on preserving the wave form. In this study, the time constant τ and bias input u_0 are the target parameters to be optimized, and others are fixed through out the whole experiments.

2.2 Dynamical Memory Layer

As has been noted, the Dynamical memory layer is expected to recall a corresponding CPG parameter $\boldsymbol{p}$ ($= [\tau, u_0, \cdots]^T$) based on the time-series observation of a system state $\boldsymbol{X}$ (e.g. $\boldsymbol{X} = [\theta, \dot{\theta}, \cdots]^T$). Therefore, it should have the following functions; 1) finding optimal CPG parameter for a motor control task, 2) the optimized CPG parameter should be stored in a dynamical memory storage for reusability purpose, and 3) estimating current situation for selecting a suitable operation mode (i.e. learn or recall CPG parameters). In the following, we explain how to implement these functions in the proposed model, which is schematically illustrated in Fig.3.

The CPG parameters to drive unexperienced controlled object are initially undetermined. In order to find better parameters, the layer has an internal learning module, where an optimal parameter $\boldsymbol{p}^*$ can be searched by using simulated annealing (SA) method[7]. Accordingly, a promising parameter at n_{th} trial $\hat{\boldsymbol{p}}_n$ is given by the following equation,

$$\hat{\boldsymbol{p}}_n = \boldsymbol{p}_{n-1} + \alpha \cdot N(0, \sigma \boldsymbol{I}), \tag{5}$$

where α is a learning rate and $N(0, \sigma \boldsymbol{I})$ represents a gaussian noise generator (here σ indicates variance of the noise and $\boldsymbol{I}$ is the identity matrix.).

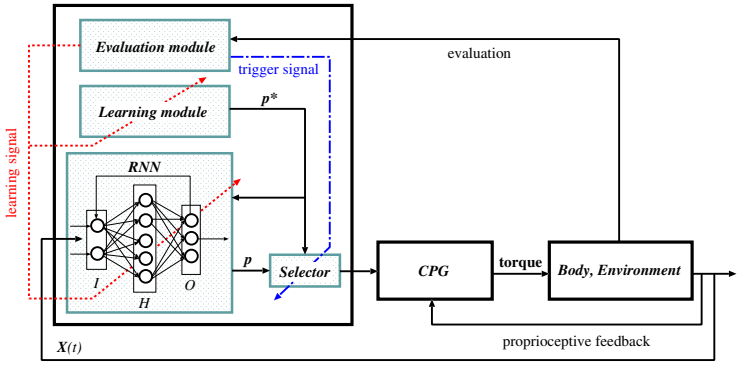


Fig. 3. Implementation of Dynamical memory layer

Now assuming that energy consumption of a controlled object at n_{th} trial is indicated by E_n (here the lower E_n corresponds to the better evaluation), the tentative parameter $\hat{\boldsymbol{p}}_n$ is accepted with the following probability P_{accept},

$$P_{accept} = \begin{cases} 1 & \cdots \text{if } E_n < E_{n-1} \\ \exp\left[-\frac{(E_n - E_{n\ 1})}{G_n}\right] & \cdots \text{otherwise.} \end{cases} \tag{6}$$

Owing to the probabilistic transition, the optimizer can avoid local minima. Moreover, a control parameter G_n, i.e. it is sometimes likened to temperature, is scheduled so as to converge the optimization process smoothly.

$$G_n = \frac{G_1}{\log n}, \tag{7}$$

On the other hand, in order to store the optimized CPG parameter $\boldsymbol{p}^*$ as the function of time-series observation $\boldsymbol{X}$, we used recurrent neural networks (RNN). The dynamics of the RNN used in this model is formulated as,

$$\tau_r \frac{ds_k(t)}{dt} = -s_k(t) + \sum_{l \in H \cup O} w_{kl} y_l(t) + \sum_{l \in I} w_{kl} X_l(t), \tag{8}$$

$$y_k(t) = g\left(s_k(t)\right),$$

$$\tilde{y}(t) = \left\{y_k(t) \mid k \in O\right\},$$

where τ_r is a time constant, w_{kl} is a connection weight, $s_k(t)$ is an internal state variable of a unit k ($k \in H \cup O$), $y_k(t)$ is the output of the unit, and $g(\cdot)$ is a sigmoid function. The subset of the RNN outputs $\tilde{\boldsymbol{y}}(t)$ can be linearly translated to the CPG parameter (i.e. $\boldsymbol{p} = A\tilde{\boldsymbol{y}}$, A is a diagonal coefficient matrix). Using a number of data consist of an optimized CPG parameter and corresponding time-series observation of the system state, the connection weight w_{kl} can be trained based on back propagation through time (BPTT) method [8].

As has been noted, there should be a system performance evaluation so that the system can select a suitable operation mode (i.e. start learning or recall stored parameter). For this aim, we also implemented evaluation module in the layer.

3 Simulation of Redundant Manipulator Control

3.1 Crank Rotation Task

Like some more complicated control problems, the proposed motion control model was applied to a crank rotation task shown in Fig.4. In this task, the

aim is to rotate the crank many times as possible during a given period by controlling a three link manipulator. As shown in the Fig.4, each joint of the manipulator is controlled by a joint torque generated by a corresponding CPG unit. Since the crank has a rotational viscous friction, in which the friction coefficient ρ can be varied by experimenter as an environmental changes, the proposed model should be able to recognize changes through the time-series observations of system states (e.g. joint angles and angular velocities) and select corresponding CPG parameters to maintain the crank rotation.

The dynamics of the crank and three-link manipulator are simulated using ODE (open dynamics engine) [9]. The Link parameters assumed in the simulation are listed in Table 1 and refers to human forearm properties. The model parameters assigned in the experiment are listed in Table 2.

In addition, the feedback term in the CPG dynamics, i.e. $F_i^{[k]}$ in equation (1) is given by the following equations:

$$\begin{aligned} F_i^{[k]} &= \kappa(q_i - q_i^e)\ (k = \text{extensor}) \\ &= -\kappa(q_i - q_i^e)\ (k = \text{flexor}), \end{aligned} \tag{9}$$

$$\mathrm{q}^e = [q_1^e, q_2^e, q_3^e]^T = [0.25\pi, 0.083\pi, \pi]^T. \tag{10}$$

The evaluation functions for crank rotation task is given by following equations:

$$E_n = -\left[E_{n_1} - E_{n_2} - E_{n_3}\right], \tag{11}$$

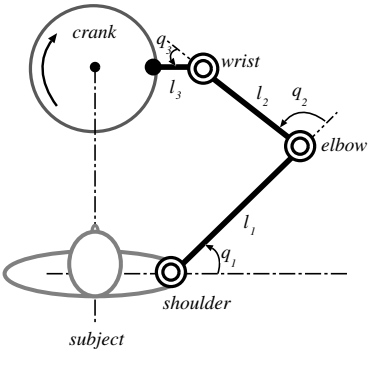


Fig. 4. A crank rotation task

Table 1. Link parameters

m_1: 1.59 $[kg]$	l_1: 0.30 $[m]$
m_2: 0.90 $[kg]$	l_2: 0.24 $[m]$
m_3: 0.54 $[kg]$	l_3: 0.11 $[m]$

Table 2. Parameters assigned in crank rotation task

N_h: # of hidden neurons	40
τ_r: time constant of RNN	1.0
α: leaning rate	0.95
σ: variance of gaussian noise	1.0
G_1: default annealing gain	10
β: fatigue constant	3.0
κ: feedback gain	1.0

$$E_{n_1} = -\frac{1}{T}\int_T \dot{\phi} dt, \tag{12}$$

$$E_{n_2} = \frac{1}{T}\int_T \frac{1}{2}\dot{\mathbf{q}}^T \mathrm{D}\dot{\mathbf{q}} dt, \tag{13}$$

$$E_{n_3} = \frac{1}{T}\int_T \max\left([\mathbf{torque}]^T C\dot{\mathbf{q}},\ 0\right) dt, \tag{14}$$

$$C = \mathrm{diag}\left[0.001,\ 0.03856,\ 34.15\right],$$

$$D = \mathrm{diag}\left[0.1,\ 0.1,\ 0.1\right],$$

where $\dot{\phi}$ is the angular velocity of crank, D is viscosity matrix for rotation. E_{n_1} represents an averaged rotation velocity, E_{n_2} indicates the dissipative energy arising from the viscous friction of each joint, and E_{n_3} corresponds to the energy externally added.

3.2 Simulation Results

Based on the above explained simulation conditions, appropriate CPG parameters for crank rotation task had been optimized.

As illustrated in Fig.5, the manipulator with optimized CPG parameters can perform skillful rotation movement. Each snap in the figure was captured in every 200 [ms]. During the first 1000[ms] (i.e. depicted A in the figure), the manipulator adjusts its posture to rotate the crank, and after the preparation, it starts periodic/sequence movements (i.e. B).

On the contrary, Fig.6 illustrates adaptation of three (i.e. shoulder, elbow and wrist) joint trajectories and torques against sudden viscous friction change (i.e. B=0.10[Nm/(rad/s)] for 0-10[sec] and B=0.15 for 10-20[sec]). According to the figures, it can be seen that the proposed model could recognize the environmental changes through proprioceptive feedback time-series and immediately select appropriate torque patterns by recalling CPG parameters. The resultant

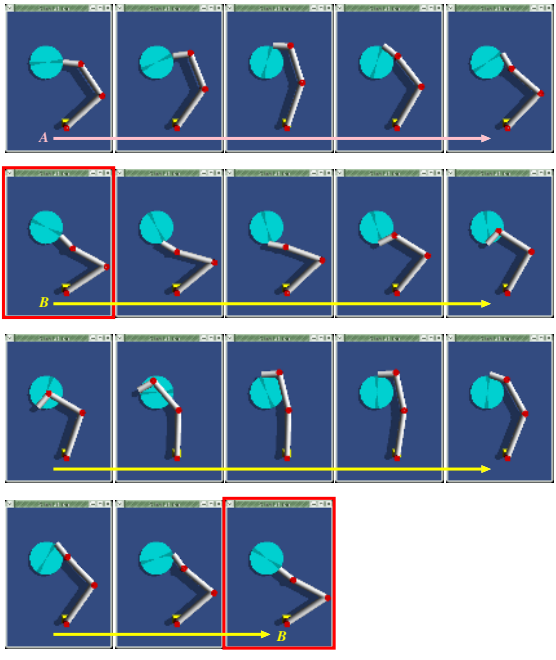


Fig. 5. Snapshots of an optimized motion (B=0.10[Nm/(rad/s)])

rotation frequency is highly depend on the strength of viscous friction. In addition, the standout gain of the wrist joint amplitude can be confirmed from the resultant trajectories after perturbations. This implies that the compliance of the wrist joint is not changed in spite of the change of angular viscous friction. Furthermore, it can be observed that the shoulder joint torque becomes larger, so as to generate larger hand torque to overcome the frictional force.

4 Conclusions

Because of limited computational resources, cerebral cortex probably does not store motion trajectories directly, rather recall some *constraint parameters* which conduct reasonable trajectories in lower motor systems so as to realize on-line adaptation. Based on this hypothesis, in the paper, we proposed a periodic motion pattern control model and its adaptation mechanism by reference to the biological brain-motor systems.

Compared with the modular selection based motor learning and control models which directly store representative motion trajectories as internal model [10,11], the proposed method can generate motion trajectories by recalling constraints. The simulation results applied to redundant manipulator control with environmental changes proved the proposed model is feasible.

Although there still exists a number of analogous motion generation model, the important issue pointed out here is that embodiment of the system

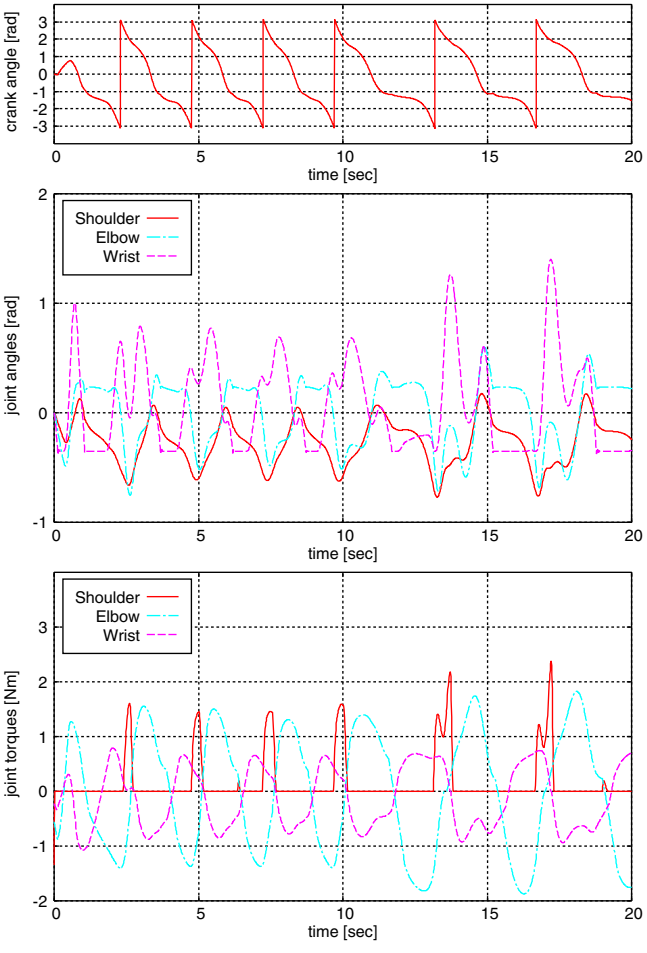


Fig. 6. Adaptation of joint angles/torques against perturbation

(i.e. dynamics of the system) has been considered in the proposed cognition/adaptation mechanism[12]. In fact, the number of stored motion patterns, i.e. attractors embedded in the RNN, is highly dependent on the pre-determined evaluation function. The proposed model expected to be useful in robotics field, since it can predictively recall a suitable parameter (i.e. motion primitive) with respect to its time-series observations, and it also can compensate external perturbation because of its dynamic entrainment ability.

Acknowledgments

This research was supported in part by a grant from the Japanese Ministry of Education, Culture, Sports, Science and Technology (No.16760337, 14350227).

References

1. Grillner, S.: Neurobiological bases of rhythmic motor acts in vertebrates. *Science* **228** (1985) 143–149
2. Yanagihara, D., Udo, M., Kondo, I. and Yoshida, T.: A new learning paradigm: adaptive changes in interlimb coordination during perturbed locomotion in decerebrate cats. *Neurosci. Res.* **18** (1993) 241–244
3. Kondo, T., Somei, T. and Ito, K.: A predictive constraints selection model for periodic motion pattern generation. Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'04) (2004) 975–980
4. Matsuoka, K.: Mechanisms of frequence and pattern control in the neural rhythm generators *Biological Cybernetics* **56** (1987) 345–353
5. Taga, G.: A model of the neuro-musculo-skeletal system for human locomotion. *Biological Cybernetics* **73**-1 (1995) 97–111
6. Williamson, M.: Neural control of rhythmic arm movements/ *Neural Networks* **11** (1998) 1379–1394
7. Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P: Optimization by simulated annealing. *Science*, **220**-4598 (1983) 671–680
8. Rumelhart, D. E., Hinton, G. E. and Williams, R. J.: Parallel Distributed Processing. The MIT Press (1988) **1**
9. Smith, R.: Open dynamics engine v0.5 user guide. (2004) `http://ode.org/`
10. Wolpert, D. M. and Kawato, M.: Multiple paired forward and inverse models for motor control. *Neural Networks* **11** (1998) 1317–1329
11. Tani, J., Ito, M. and Sugita, Y.: Self-organization of distributedly represented multiple behavior schemata in a mirror system: review of robot experiments using RNNPB. *Neural Networks* **17** (2004) 1273–1289
12. Pfeifer, R. and Scheier, C. Eds.: Understanding Intelligence. The MIT Press (1999).

Self-organized Criticality on Growing Scale-Free Networks

Yuumi Kawachi[1] and Shinichiro Yoshii[2]

[1] Hokkaido University, Sapporo, Japan
kawachi@complex.eng.hokudai.ac.jp
[2] Hokkaido University, Sapporo, Japan
yoshii@complex.eng.hokudai.ac.jp

Abstract. This paper explores a universal property in the behavior of growing scale-free networks. The characteristic of scale-free networks is that the degree distribution follows the power-law. This structure has been found in various kinds of self-organized networks. Most investigations conducted so far have demonstrated that network topologies are scale-free at a specific point in time. On the other hand, we focus attention on universality in the growing process of networks. In our proposed model, each node has its own fitness to designate the tendency allowing the node to acquire new links. From the simulation results, spread of the network follows the power-law, and power spectrum of the growing process shows $1/f$ noise, not to mention that the network has scale-free structure. It is found that those properties are in common with self-organized criticality. In conclusion, self-organizational growing networks follow the power-law not only in the sense of scale-free characteristic but also in the spatial and temporal sense.

1 Introduction

Self-organization means that structures of some kind appear without explicit pressures or constraints from outside the system. For complex systems, where constituent elements interact with each other, self-organization is an extremely important concept. Additionally, the interaction that emerges is more than a mere aggregation of the elements. A number of researchers have studied self-organization, for example galaxies, planets, compounds, cells, organisms and societies. Par Bak, in particular, has put forward the idea of criticality in self-organization. Par Bak has illustrated common phenomena in critical states seen in various natural systems [1].

In recent years, networks or network structures have attracted much attention, as exemplified in [2], [3], [4]. One of the reasons for this is that the Internet and World Wide Web have become more popular and closer than before in everyday life. Moreover, although they are constructed by human beings with human intentions, and these networks seem to be random because of their large scale and complexity, it has been found that they have the common structures with some other networks in nature.

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 916–925, 2005.

We consider now the implications of "networks" composed of elements and their interactions in a system. Networks, which are fundamental to the universe, include complex systems. Therefore, some networks would show self-organization or self-organized criticality in complex systems.

1.1 Universality in Self-organized Networks

As described above, the Internet, World Wide Web, and some other natural or artificial networks, for instance, metabolic network, human relations, airline route etc. have a common structure termed "scale-free". The most remarkable feature of scale-free networks is seen in degree distribution. The number of nodes that are connected to k links follows the power-law. That is, the probability $P(k)$ that one node connects to other k nodes is proportional to $k^{-\gamma}$, where γ is the number of connected links from other nodes.

$$P(k) \sim k^{-\gamma} \tag{1}$$

As can be expected from the Eq.(1), a generic graph that indicates the relation between two parameters of scale-free networks, the natural logarithm of the number of links k and that of the degree distribution $P(k)$, shows an almost straight line.

Most previous studies have demonstrated that network topologies at a specific point in time are scale-free. Little is known about the growing process of networks. We consider that the growing process has universality whether the networks exist in nature or are constructed by artificial means. We would like to point out a universal property in the behavior of growing scale-free networks from the viewpoint of phenomena observed in nature. The purpose of this study is to verify that self-organized criticality is observed in the formation process of scale-free networks.

2 Self-organized Criticality

Self-organized criticality is the boundary between order and chaos where a large system, in which many elements are intricately interrelated, moves spontaneously without external control. The idea of a critical state is clear, but there is no rigorous definition. In some articles, 'power-law behavior without parameter fine tuning' or 'scale invariance' or 'power-law distributed events' is described as a definition [5].

However, Per Bak is the most famous researcher who established the idea of self-organized criticality. Per Bak *et al.* [1], [6], [7] state that certain extended dissipative dynamical systems naturally evolve into a critical state, with no characteristic length or time scales. That is, spatial signature is the emergence of scale-invariant structure and the temporal signature of the self-organized critical state is the presence of $1/f$ noise. The critical states system has the property in which the power-law emerges in a spatial and temporal manner. Further studies

[8], [9] show the spatial and temporal power-laws as the critical states. Therefore, we use this definition in this paper. In the real world, for example, sand piles that are described below, the fractal structure of river networks, turbidities, earthquakes, starquakes, solar flares and so on have self-organized criticality.

Next, we will introduce two major models for self-organized criticality phenomena in order to compare with our proposed model in section 3. The main theoretical issues to be addressed are that these models organize themselves to the critical state, which is characterized by the power-law.

2.1 Sand-Pile Model

When sand trickles down, a pile is formed on the ground. In the beginning, as the process continues, the pile becomes steeper. Eventually, the pile has a constant slope while there are large and small sand slides. The state in which the system maintains the structure with recurrent avalanches is self-organized criticality.

The model which expresses the forming sand pile is simulated in a two-dimensional grid ($L \times L$). One grain of sand is dropped at the grid randomly by one step. The grid is represented by (x, y), where $1 \leq x \leq L$ and $1 \leq y \leq L$, and the number of grains which are dropped in the grid is represented by $Z(x, y)$. That is

$$z(x, y) \rightarrow z(x, y) + 1 \tag{2}$$

Next, let us consider the avalanches of a sand pile. When the number of grains at one grid exceeds the critical value Z_{cr} $(= 3)$, in other words the number reaches four, one grain of sand is sent to each of the four neighbors. Fig. 1 shows this model.

$$Z(x, y) \rightarrow Z(x, y) - 4 \tag{3}$$

The four neighbor sites go up by one as follows:

$$\begin{aligned} Z(x \pm 1, y) &\rightarrow Z(x \pm 1, y) + 1 \\ Z(x, y \pm 1) &\rightarrow Z(x, y \pm 1) + 1 \end{aligned} \tag{4}$$

In this model, the number of influenced grids by one grain drop indicates the size of the avalanches. The distribution of the size of avalanches follows the power-law at the critical state. Moreover, in any grid, the power spectrum of the number of changes in $Z(x, y)$ for time steps follows $1/f$ noise. That is to say, the Sand-Pile model follows the spatial and temporal power-law.

2.2 Evolutionary Model

The basic idea for evolution is that the species with lowest fitness is inclined to disappear or mutate in environments. A simple model of evolution, which is associated with punctuated equilibrium and criticality is described in [10]. In this model, uniform random numbers between 0 and 1 are arranged for each species

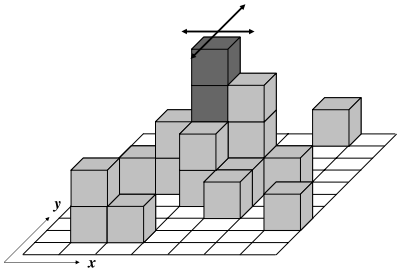


Fig. 1. Sand-Pile Model: one unit is a grain of sand. When the number of grains exceeds Z_{cr} at one grid, the grains are sent to each of the four neighbors.

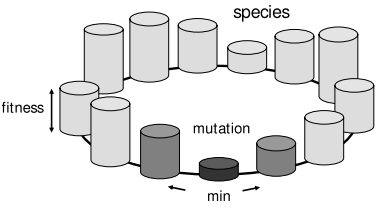


Fig. 2. Evolutionary Model: a species with the lowest fitness and the two neighbors are mutated

as fitness. Each of them is on the circle and interacting with its two neighbors. At every time step, the species with lowest fitness and its two neighbors mutate because of the interaction. Mutation means that each fitness of these three species is replaced by new uniform random numbers between 0 and 1. This model is shown in Fig. 2.

Although each species' fitness fluctuates up and down, the average tends to increase to a certain value. The envelope function of the lowest fitness, defining the fitness gap, increases in a stepwise manner. Fitness gap is the difference between the lowest and that of the one higher than it. Furthermore, the size of avalanches is the length of the step while the fitness is unchanged, that is, the number of mutations. When there is a gap, a new avalanche begins. There are avalanches of all sizes; that is to say the distribution of the size of avalanches follows the power-law. In the same manner as the Sand-Pile model, the power spectrum of the cumulative number of mutations for any species at critical state follows $1/f$ noise. The evolutionary model also follows the spatial and temporal power-law.

3 Growing Scale-free Networks

Albert-László Barabási [11] has suggested that the formation of scale-free network needs preferential attachment. "The rich get richer" mechanism makes the property of scale-free structure related to degree distribution. This mechanism, however, does not take into consideration the fact that each node has different characteristics or properties. In the real world, nodes that form some networks are not equal. For example, all web pages on the World Wide Web are different. In this paper we propose a new model that takes into account each node's fitness, which indicates the variations in every node. As a result of the simulation experiments, it is found that scale-free networks are formed not only by the degree of nodes, as in Barabási's study, but also by the fitness for each node. In addition,

we investigate the spread of the network and the growing process in time taking into account the fitness change for self-organized criticality.

3.1 Fitness Growth Model

The proposed model is the *Fitness Growth Model*. Our preferential attachment depends only on node fitness. A node with a higher fitness has a higher probability of acquiring new links. Moreover, the node which gets a new link increases its fitness depending on the new node's fitness. The update equations for a new link and fitness are represented by Eq. (5) and Eq. (6). A new node is added to the network with two links as in the BA model [11] at one step over and over again. This idea is represented in Fig. 3,

$$E_{t+1}(i,j) = V_t(j) \cdot \frac{f_j(t)}{\sum_m f_m(t)} \tag{5}$$

$$f_j(t+1) = f_j(t) + \alpha \cdot f_i(t) \tag{6}$$

where $E_t(i,j)$ is link existence probability between nodes i and j at time t, $V_t(j)$ is existence probability for node j at time t, $f_i(t)$ is the fitness for node i at time t and α is a positive parameter, in this paper $\alpha = 0.5$. The initial value of f is chosen from a uniform random number between 0 and 1. Our proposed model includes BA model that related to the number of links for each node when initial value $f = 2$ and parameter $\alpha = 0.5$. Node fitness in our model represents not discrete number such as natural number but more detailed information to investigate the data for self-organized criticality.

Figs. 4–7 show the degree distribution of the growing network in this model. As we can see from these figures, the network has scale-free structure and maintains the structure during the growth. The shapes of network structures appear

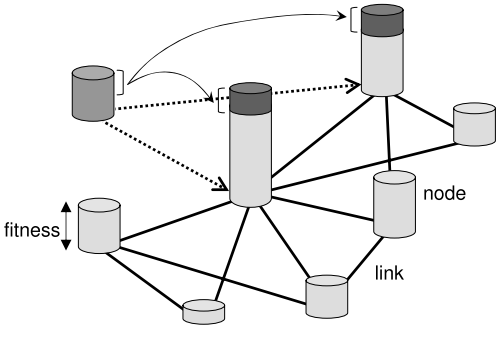


Fig. 3. Fitness Growth Model: when a new node attaches to two other nodes, their fitness increase according to the new node's fitness

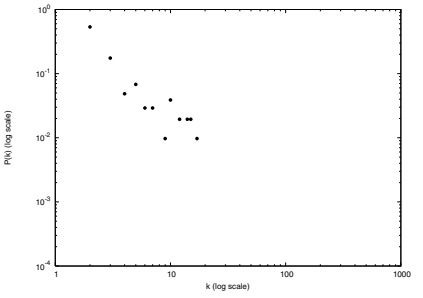


Fig. 4. Degree distribution at 100th step

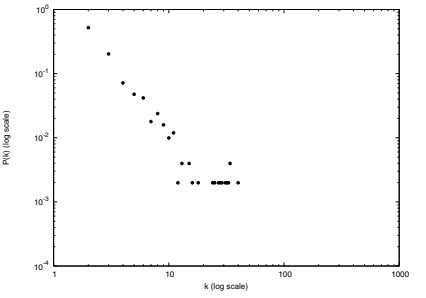


Fig. 5. Degree distribution at 500th step

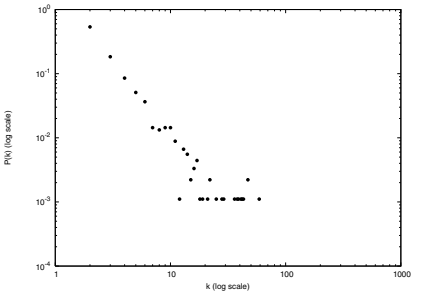


Fig. 6. Degree distribution at 900th step

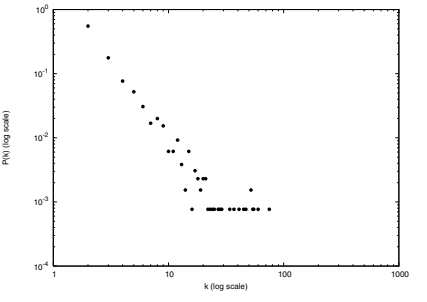


Fig. 7. Degree distribution at 1300th step

in Fig. 8. Therefore, it is clear that we can simulate a growing scale-free network with our fitness model. We shall analyze the spread of the network and the transition of node finesses to explain self-organized criticality phenomena.

3.2 Model Comparison

To explain critical phenomena in our model, it is necessary to compare with other models that indicate self-organized criticality. Table 1 shows the models and conditions for the critical phenomenon. First, there exists a critical point or critical state where the system maintains equilibrium. In the case of the Sand-Pile model, it is the constant slope of the sand pile and in the case of the Evolutionary model, it is the minimum fitness in which almost all the species possess an excess. When we consider a growing scale-free network with the fitness model, structure preservation is the critical state. Even though the network is growing, it maintains the scale-free structure with the constant slope of the degree distribution graph.

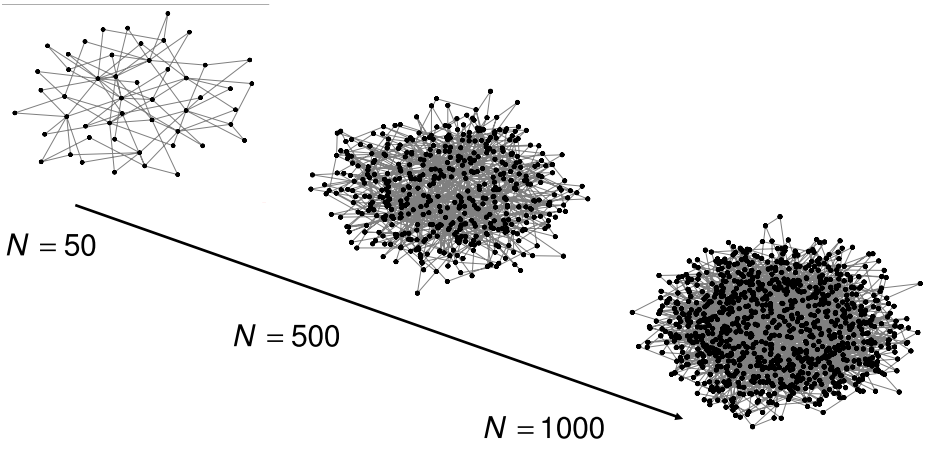


Fig. 8. Growing scale-free network with fitness growth model. N is the number of nodes.

Next, we must investigate whether the growing scale-free network follows the spatial and temporal power-law as described in the previous section. Par Bak suggests that the size distribution of avalanches shows spatial power-law in the Sand-Pile model and Evolutionary model. The size of avalanches refers to the span of punctuated equilibrium, and during the span the elements in the system interact with each other for the equilibrium. In the case of our fitness model, the constant spans of maximum fitness represent the avalanche, because other elements are added and the network is growing during the span to maintain the structure. Maximum fitness is an index for expanding the whole network because the straight line of degree distribution extends in a direction toward large fitness nodes as Figs. 4–7. After the network grows to an extent, it is found that the same node always has the maximum fitness.

In addition to spatial signature, temporal signature is the transition of any node's fitness in our model. If the power spectrum of the transition shows $1/f$ noise, it indicates the critical state. In the Sand-Pile model and Evolutionary model, temporal signatures at critical states emerge in the transition frequency at one site or species. That is, to keep the equilibrium of the whole system, the behavior of one element in the system shows temporal signature for critical states. It is considered that the signature has a characteristic power-law $1/f$ behavior. Therefore, we analyze the change of any node's fitness that is chosen randomly. Especially, in order to get enough data for spectral analysis we choose a node which exists more than 2000 steps.

Here we only use a model that constructs scale-free structure. What happens with the growth of a random network? With the Erdös-Rényi random model [12], even if we get the same data as the scale-free network model, maximum fitness or one node's fitness, it is found that these data would not follow any power-law. Although we would not show the results, nodes are added to the network randomly in this case and the fitness for each node does not

Table 1. Model comparison with self-organized criticality

	Fitness Growth Model	Sand-Pile Model	Evolutionary Model
Critical state	Constant slope of degree distribution.	Constant slope of sand pile.	Constant value of minimum fitness.
Special signature	Span distribution of maximum fitness.	Size distribution of avalanches.	Size distribution of avalanches.
Temporal signature	Transition of a node fitness.	Transition of avalanche's cumulative frequency at one site.	Transition of mutation's cumulative frequency at one species.

increase in a staircase pattern such as the case of scale-free network. The reason why we focus on scale-free structure is that the structure is under the power-law.

On the grounds that critical states follow the spatial and temporal power-law, such as in Table 1, we conduct simulation experiments with our fitness growing network model. In the next section, some results are shown.

4 Spatial and Temporal Power-Law

As described above, Figs. 4–7 show the degree distribution in which a scale-free network maintains the structure during the growth. Then, the transition of maximum fitness in the whole network shown in Fig. 9 and Fig. 10 is the result of the size distribution in Fig. 9 from step 1 to step 15000. The size of one span is where the maximum fitness is constant between a step and the next step in the graph. The size distribution is associated with the power-law, though there has been no generic measure of how much of a power-law the network

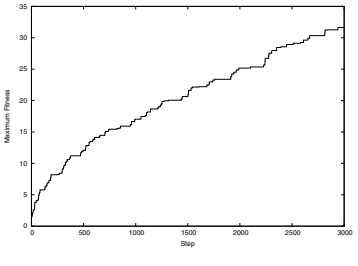


Fig. 9. The maximum fitness transition of growing scale-free network

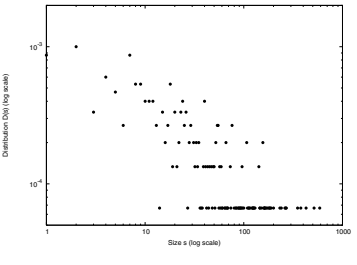


Fig. 10. Spatial Power-Law: The distribution of the step length which is constant in Fig. 9 from step 1 to step 15000. This graph has a correlation with power-law.

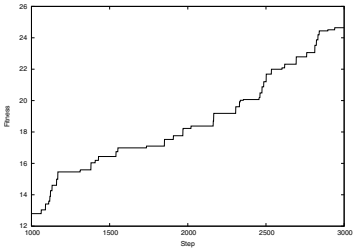


Fig. 11. Transition of a node's fitness during criticality, from step 1000 to step 3000

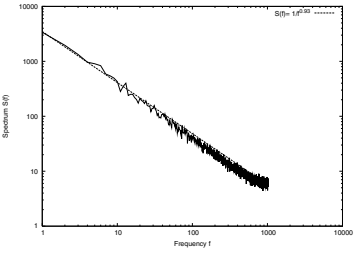


Fig. 12. Temporal Power-Law: Power Spectrum of Fig. 11 shows $1/f$ noise

is. We consider that the result indicates the spatial signature at critical state. Moreover, the transition of one node's fitness during the critical state (from step 1000 to step 3000) is shown in Fig. 11. When this transition is analyzed with Fourier analysis, we get the power spectrum as shown in Fig. 12. The power spectrum provides $1/f$ noise, which is approximated by $S(f) \sim 1/f^{0.93}$ in our simulation.

From the results of spatiotemporal signature, it is found that growing scale-free networks indicate self-organized critical phenomena. One of the reasons why there exist scale-free structures in many fields in the real world is that common mechanisms that are explained by natural phenomena are at work in these networks.

5 Conclusion

As consequences of simulation experiments, it is found that growing scale-free networks have self-organized criticality because of the spatial and temporal power-law. We use a simple model for growing networks; therefore, it becomes the basis of the behavior of all scale-free networks. It follows from this that a universal property exists in "networks" around us. In particular, we treat the process of growing networks not only from network topology at a specific point in time.

In closing, it is necessary to research and develop new information systems using such universal properties associated with networks. We call this approach "Hyper Net Intelligence", the idea of which is to construct information processing systems with intelligence hidden in network properties. With this approach, we aim to develop Web applications in the future.

References

1. Per Bak: How Nature Works: The Science of Self-Organized Criticality. Copernicus Books (1999)
2. Duncan J. Watts: Small Worlds: The Dynamics of Networks Between Order and Randomness. Princeton University Press (1999)
3. Mark Buchanan: Nexus: Small Worlds and the Groundbreaking Science of Networks. W. W. Norton & Co Inc (2002)
4. S. N. Dorogovtsev, J. F. F. Mendes: Evolution of Networks: From Biological Nets to the Internet and WWW. Oxford University Press (2003)
5. Roman Frigg: Self-organised criticality — what it is and what it isn't. Studies in History and Philosophy of Science A **34** (2003) 613–632
6. Per Bak, Chao Tang, Kurt Wiesenfeld: Self-organized criticality. Phys. Rev. A **38** (1988) 364–374
7. Per Bak, Chao Tang, Kurt Wiesenfeld: Self-Organized Criticality: An Explanation of $1/f$ Noise. Phys. Rev. Lett. **59** (1987) 381–384
8. Deepak Dhar: Self-organized Critical State of Sandpile Automaton Models. Phys. Rev. Lett. **64** (1989) 1613–1616
9. Yang, C.B.: The origin of power-law distributions in selforganized criticality. J. Phys. A **37** (2004) 523–529
10. Per Bak, Kim Sneppen: Punctuated Equilibrium and Criticality in a Simple Model of Evolution. Phys. Rev. Lett. **71** (1993) 4083–4086
11. Albert-László Barabási: LINKED:The New Science of Networks. Perseus Publishing (2002)
12. Réka Albert, Albert-László Barabási: Statistical mechanics of complex networks. Reviews of Modern Physics **74** (2002)

Synapsing Variable Length Crossover: An Algorithm for Crossing and Comparing Variable Length Genomes

Ben Hutt and Kevin Warwick

Department of Cybernetics, School of Systems Engineering, University of Reading, Whiteknights, Reading, Berkshire, RG6 6AY, UK
{B.D.Hutt, K.Warwick}@rdg.ac.uk

Abstract. The Synapsing Variable Length Crossover (SVLC) algorithm provides a biologically inspired method for performing meaningful crossover between variable length genomes. In addition to providing a rationale for variable length crossover it also provides a genotypic similarity metric for variable length genomes enabling standard niche formation techniques to be used with variable length genomes. Unlike other variable length crossover techniques which consider genomes to be rigid inflexible arrays and where some or all of the crossover points are randomly selected, the SVLC algorithm considers genomes to be flexible and chooses non-random crossover points based on the common parental sequence similarity. The SVLC Algorithm recurrently "glues" or synapses homogenous genetic sub-sequences together. This is done in such a way that common parental sequences are automatically preserved in the offspring with only the genetic differences being exchanged or removed, independent of the length of such differences. In a variable length test problem the SVLC algorithm is shown to outperform current variable length crossover techniques. The SVLC algorithm is also shown to work in a more realistic robot neural network controller evolution application.

1 Introduction

Traditional Genetic Algorithms operate on a population of fixed length genomes, which are viewed as a set of potential solutions to a problem. Typically, the genome is regarded as a set of variables, which are to be optimised by the GA in order to solve a given problem. In other words the goal of the GA is to find an optimum within a given, fixed parameter space. If a large number of parameters are required to be optimised then the search space a GA must traverse in order to find the optimum can be immense. Considering the length of biological genomes a classical GA is unlikely to be a good model of the long term process of biological evolution.

The human genome, for example, is vast and consists of approximately three billion base pairs specifying some hundred thousand genes [1]. In fact, if the DNA from a single human cell were to be unravelled and stretched out it would be nearly two metres long. It would clearly be preposterous to think that evolution had started with strings of this length and merely optimised them over time. Instead it is thought that life started off rather more simply with complexity arising over time. First, single celled organisms evolved, these then led to multi-cellular organisms, which in turn led

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 926 – 935, 2005.

to the first plants and animals. Complexity and genome length has been able to grow gradually over several billion years through the course of natural evolution. It is worth noting that biological evolution does not necessarily cause an increase in complexity or genome length both can change or remain static over time, all that matters is how good an organism is at passing on its genes to subsequent generations. Also, a longer genome does not necessarily lead to increased complexity. It is only *possible* for complexity to increase, whether this occurs or not is likely to be dependent on the organism in question and the environmental conditions.

2 Variable Length Genomes

A number of studies have investigated variable length genomes in the context of Genetic Algorithms. In all of the methods described below, a position independent genotype encoding must be used. This can be achieved by either encoding the locus in a fixed length representation or by letting certain genetic sequences acts as markers, that effectively identify the sections encoding problem data – see [2]. Other evolutionary methods such as Koza's Genetic Programming [3] and certain Evolution Strategies [4] also use variable length representations. However, the investigation presented in this paper is limited to the context of Genetic Algorithms as other non-GA based variable length evolutionary methods differ in fundamental ways to those used by Genetic Algorithms. For example, Genetic Programming techniques evolve programs (such as Lisp S-expressions), whilst Evolution Strategies use adaptive mutation rates and rather different selection strategies, making them exceedingly difficult to compare to any GA based methods in a consistent manner.

2.1 Messy Genetic Algorithms

One of the first studies to utilise a variable length representation was the messy GA [1]. In a messy GA the traditional crossover operator is replaced by the cut and splice operators. First the cut operator is applied on each parent genome, a point is selected at random on each genome cutting each genome into two strings, forming four strings. Then the splice operator is applied to rejoin the strings in a random order.

Whilst messy GAs use a variable length representation they are in fact still based on a fixed length scheme. The reason for this is that genes in a messy GA contain both a value and a tag that specifies the position or locus of that value in a fixed length genome. The messy GA allows for both over specification and under specification of the fixed length genome. In the case of over specification, there are multiple specified values for a specific locus. In this case an ad hoc rule is used to decide which value gets to be used at the specified locus – a popular method is to give precedence to the last gene in the genome. An easy way of implementing this is to allow specified values to overwrite those contained in the fixed length genome, thereby giving precedence to the last value specified. In the case of under specification, some loci in the fixed length genome do not have a specified value. For such loci with missing values the value is taken from a universal template in order to fill in the gap – this template would usually contain a sensible initial or default value for that particular variable.

2.2 The SAGA Cross

In [5] the concept of the SAGA cross is introduced. The idea of this algorithm is to maximise the similarity between sections that are exchanged during recombination thereby ensuring that a sensible and meaningful crossover occurs between the parent genomes. The SAGA cross is intended to be used within the general SAGA framework.

In a fixed length GA, crossover is trivial as the crossover point on both parents is at the same position - the offspring of such a cross will be of exactly the same fixed length as that of the parents thereby ensuring that each individual has a full complement of genes. Within the SAGA framework the evolving population is relatively converged - meaning that different values at the same locus are highly likely to reflect different values of the same phenotypic variable.

The question is how crossover should be applied to variable length genomes in order to ensure that resulting offspring have a full complement of genes and to ensure that only the values of like phenotypic variables are exchanged.

The SAGA cross is based on the sequence similarity of the two parent genomes to be crossed – the metric used is the Longest Common Sub Sequence. The Longest Common Sub Sequence is the longest uninterrupted matching sequence of symbols found between two strings of arbitrary length. Meaning is therefore given by the context of a gene within the genome. If a gene has the same context it is likely to be a different allele of the same gene.

The SAGA algorithm works as follows: first a random crossover point is chosen on the first genome, then the algorithm tests every possible crossover point on the second genome. For each potential crossover point the algorithm calculates the sum of the LCSS on both the left and right portions of the genomes. Only the crossover point(s) with the highest score are eligible as a crossover point for the second genome. If there are multiple eligible crossover points then one of them will be selected at random.

2.3 Virtual Virus

A slightly more biologically plausible crossover is the homologous crossover used by the virtual virus (VIV) project [6]. Like the SAGA cross the VIV crossover algorithm is based on the sequence similarity between parent genomes. The major difference from the SAGA cross is that crossover can only occur in similar sections. In VIV the probability of crossover is controlled by the degree of local similarity between the parent genomes. This local similarity is determined by the number of matches between parents within a specified fixed size window.

The VIV algorithm works as follows: as in the SAGA crossover algorithm a random crossover point is initially chosen on one of the parent genome. The algorithm then compares a window of bases from the selected point with all possible windows of the same size on the other parent genome. The window position that achieves the greatest number of matches is then recorded. The genomes are then crossed within the matched window with a probability based on the similarity of the best matching window.

2.4 Overview of Current Methods

All the above methods provide a rationale to perform Variable Length Genetic Crossover, messy GAs use a simple cut and splice implementation – however this totally ignores any sequence similarity between parent genomes and worse still, messy GAs are fundamentally based on an underlying fixed length representation. Both the SAGA and VIV crossover methods are based on the similarity between the two parent genomes. Both select a random point on one genome and then search the entirety of the other parent genome in order to find the optimal crossover point(s). The SAGA cross however explicitly attempts to preserve a complete genetic sequence using the parent genomes as a guide and is, in this respect, far superior to the VIV cross. It is important to note that in all these variable length methods the genomes are considered to be inflexible, rigid arrays of data. In addition the selection of the initial crossover points are entirely random – this is unlikely to be the case in biological crossover.

3 Biological Crossover Revisited

In animals genetic recombination by crossover occurs during the production of sperm or egg cells (*gametes*) this process is called *meiosis*. Before meiosis begins the chromosomes within the cell have already been doubled, forming pairs of homologous dyads. Each dyad consists of two sister chromatids held together by a single shared kinetochore. First, each pair of homologous dyads align lengthwise with one other and form synapses between themselves resulting in a tetrad. The two homologous dyads are held together at one or more chiasmata. The chromatids attached by chiasmata then slip apart and the homologous dyads separate from each other.

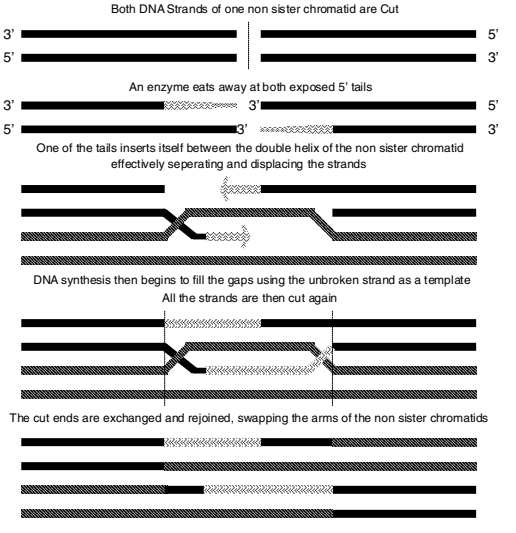


Fig. 1. A plausible mechanism for crossover

Chiasmata are in fact the remnants of where non-sister chromatids swapped sections whilst they were joined together during the early stages of meiosis. The process of crossover effectively exchanges adjacent strands of DNA. However, this exchange must be performed in such a way that not even a single nucleotide is lost or gained during the exchange. The exact mechanism of crossover is unknown but a plausible mechanism that has significant support is as follows [7]. During the process of crossover one non-sister chromatid is cut through on both strands and an enzyme eats away at these exposed ends leaving two single stranded tails. One of these tails then invades the double helix of the other non-sister chromatid displacing the strands. The invading strand then aligns itself with a complementary sequence of nucleotides with which it can pair; the complementary tail also pairs up but with the other displaced strand. DNA synthesis then fills in any gaps using the second unbroken chromatid as a template. Then all the strands are cut, exchanged and rejoined, effectively exchanging the arms of the non-sister chromatids, this process is illustrated in Fig. 1.

4 Inspiration from Biological Crossover

It is worth noting that crossover can only occur in regions that are homogenous due to the double stranded structure of DNA, consequently each strand of DNA can only pair with a complementary strand. Fig. 1 only shows a single chiasma however multiple chiasmata are commonly found.

We can therefore safely assume that, in general, crossover does not occur between regions that have a dissimilar sequence and that it is highly likely to occur between regions that have identical or a highly similar sequence. More importantly we should consider the regions between chiasmata – it is these sections that are likely to contain any significant genetic differences between genomes and it is these sections that are actually exchanged by crossover. Further, we should consider the fact that DNA is a flexible molecule and so the sections actually exchanged may be of differing length.

Thus, by restricting crossover to homogenous sequences the common sequence similarity is preserved using both parent genomes as a template and only the differences between the parent genomes are exchanged independent of the length of those differences. Intuitively, it must be the differences between genetic sequences that cause any variation in phenotypic fitness. Thus by preserving the similarity and exchanging differences an efficient crossover is performed, therefore only genetic variations that may cause phenotypic fitness variations are recombined.

5 SVLC Algorithm

In order to perform such a crossover we must first have a method with which to compare the two genetic sequences for similarity in order to find regions, which are to be subsequently synapsed together. The metric used is the same as that used by Harvey's SAGA crossover algorithm – the Longest Common Sub Sequence. In addition Harvey also points us towards an efficient algorithm for finding the LCSS. A modified version of this algorithm is used to recursively find the LCSS in order to find any similarities between genomes with priority given to longer common sub sequences. Full

C++ source code to perform this crossover is available by e-mailing the author at [B.D.Hutt@rdg.ac.uk].

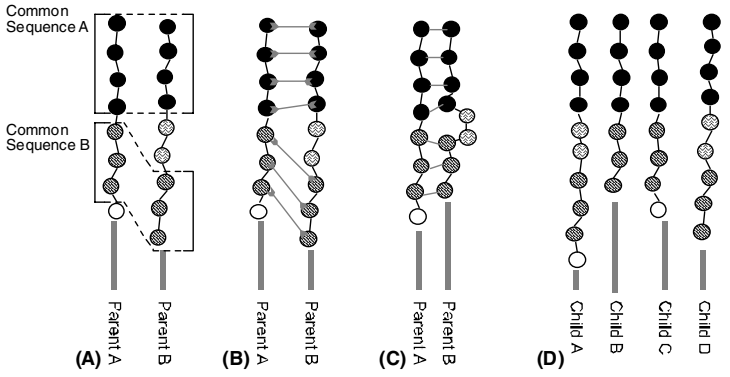


Fig. 2. Example of Synapsing Variable Length Crossover. (A) Two similar parent genomes are shown; The common sequences on the parent genomes are labelled. (B) First regions that are identical are identified. (C) The genome is thought of as being flexible so that identical sections can be aligned - crossover is only permitted within these identical regions. (D) The implication of this is that the similarity between parent genomes is always preserved – it is only the differences between parent genomes that are exchanged.

The rationale behind the algorithm is as follows, given two variable length parent strings the position and length of the LCSS between the two strings is located.

The position and length of the LCSS is recorded and the beginning and end of the LCSS on *both* strings is then used to create two sub problems, finding the LCSS between the two sub strings to the left of the current LCSS and the two sub strings to the right of the current LCSS. This is repeated recursively only dropping out when the LCSS is shorter than a predefined limit. This process produces a list of matched or synapsed segments between the two parent strings. This list is then collated in order to form a set of possible crossover points between the two genomes.

Crossover can then be performed by choosing one or more crossover points at random from this set of possible crossover points in order to produce the offspring.

This process is illustrated in Fig. 2, the two parent genomes are effectively synapsed together at their points of similarity, with longer common sub-sequences being given priority. This aligns the chromosomes in a sensible fashion with identical regions being synapsed together. Crossover is then only permitted within the synapsed regions producing offspring that inherit the entirety of the common sub sequences of the parents. Any sequence differences between the parent genomes may or may not be included in the offspring dependent on the crossover point(s) selected.

5.1 Computational Requirements

Since the SVLC (Synapsing Variable Length Crossover) algorithm recursively eliminates the LCSS from each string it will always be somewhat less computationally efficient than the SAGA crossover algorithm. The SAGA crossover algorithm

chooses a one-point crossover between parent strings of length m and n, this is done extremely efficiently, being of the order O(mn) having the additional advantage of being time independent of the sequence similarity between parent genomes [5].

The SVLC algorithm is however highly dependent on the similarity between the parent strings. At worst case the SVLC algorithm will always be less than order O(mn^2) however this is a worst case and things are exceedingly unlikely to be this bad, especially when we consider that evolving populations within a SAGA framework are considered to be relatively homogenous [8]. In the best case the algorithm will be as efficient as the SAGA cross and be of order O(mn), this is when both parent genomes are identical or entirely dissimilar. Since we can expect the population to be relatively converged we can also expect that the order will be much closer to order O(mn) rather than O(mn^2).

Due to the fact that SVLC becomes increasingly computationally expensive the longer a genome grows, it is intended for use primarily when fitness evaluation is also computationally expensive, such as is the case when evolving neural network controllers for robots. If fitness evaluation is not computationally expensive and a large number of generations are required and the resulting genomes are likely to be excessively long, then another variable or fixed length method should be considered.

5.2 Similarity Metric

In addition to locating crossover points the SVLC algorithm can be used to give an effective metric with which it is possible to compare the similarity of two variable length strings. This is given by:

$$H = \frac{2\sum_{i=0}^{\psi-1} L_i}{m+n}$$

Where, L_i is the length of the ith synapsed section, ψ is the total number of synapsed sections and m and n are the lengths of the two parent genomes. H therefore gives us a hamming distance like measure of the similarity or homogeneity of the two genomes and varies between *0.0*, for genomes that are totally dissimilar and *1.0* for identical genomes. This immediately provides a method by which standard fixed length genetic algorithm niche formation techniques can be directly applied to variable length genetic algorithms.

6 Initial Results

In order to test the performance of the SVLC algorithm it was tested on a simple variable length problem. The task was to produce a piecewise linear approximation of a non-linear target function *T(x)* given by:

$$T(x) = 0.5(1+\sin(20.\pi.x)) \text{ where } 0 \le x \le 1$$

Genotypes are interpreted as a set of points between which lines are drawn in order to create the actual phenotype *P(x)*. The fitness F is calculated using an estimate of the root-mean square error being given by:

$$F = 1 - \sqrt{\frac{1}{N}\sum_{x=0}^{1}\left(T(x) - P(x)\right)^2} - \mu.L$$

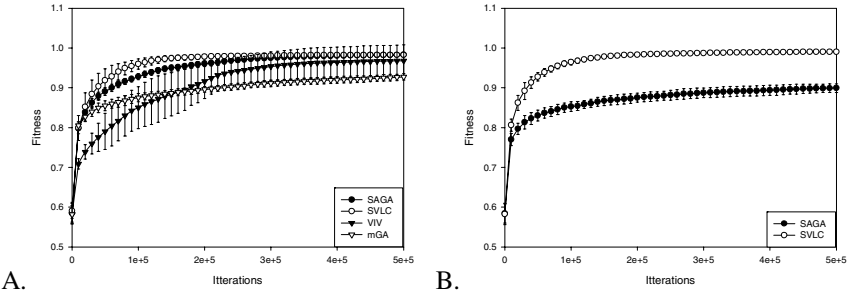


Fig. 3. (A) Comparison of various crossover algorithms with a length penalty in place. (B) Comparison of SAGA and SVLC crossover algorithms with no length penalty in place.

The RMS error is estimated by the sum over N discrete parts, with the constant μ effectively scaling the genome length L to create a weak genome length penalty. In all cases Harvey's microbial GA [9] was used with a tournament size of 2 and an initial genome length of 32 with each crossover algorithm being tested over 10 runs of $5x10^5$ iterations.

The first set of tests are shown in Fig. 3A with the length penalty μ set to 10^{-6}. As can be seen both the SAGA and SVLC crossover algorithms rapidly obtain excellent approximations of the target function. The VIV crossover algorithm takes far longer to obtain a good solution and the messy GA struggles to obtain a good solution. Fig. 3B shows a similar test run for both the SAGA and SVLC crossover algorithms but with the length penalty μ set to 0. Surprisingly this cripples the performance of the SAGA crossover to a similar level to that of the messy GA in the previous run whilst the SVLC algorithm remains unaffected. Without a length penalty in place it is possible for fitness neutral sequences to increase in length. It therefore seems likely that the SAGA crossover can be misled by long neutral sequences forming the LCSS. In such circumstances the SAGA algorithm will select crossover points based on neutral portions of the genome. SVLC does not suffer from this deficiency since it recursively synapses common sequences together restricting crossover to such sections. It is worth noting that testing on a far wider range of variable length test problems is required to properly evaluate the performance of the SVLC algorithm.

7 Neuro-controller Evolution Results

In order to further test the effectiveness of the SVLC algorithm it was applied to a more difficult behavioural evolution task. An extended GA utilising the SVLC algorithm was employed to evolve spiking neural network controllers for simulated robots in a simple 2D environment. The environment consisted of an otherwise

empty square enclosure containing the entire population of 100 simulated robots. The robots themselves are of a differential drive type (similar to those described in [10]) and are equipped with their own genetically determined spiking neural network controller and forward facing proximity sensors. In order to encourage an open-ended wandering/exploratory behaviour an *implicit* fitness metric was utilised. The fitness of each individual was a measure of the *unique* area a robot is able to cover in a fixed period of time. This was implemented by allowing each robot to collect "food" pixels as it moved around the enclosure, once a pixel has been collected it changes state and cannot be collected again, the fitness of each robot was therefore the number of pixels collected during the evaluation period.

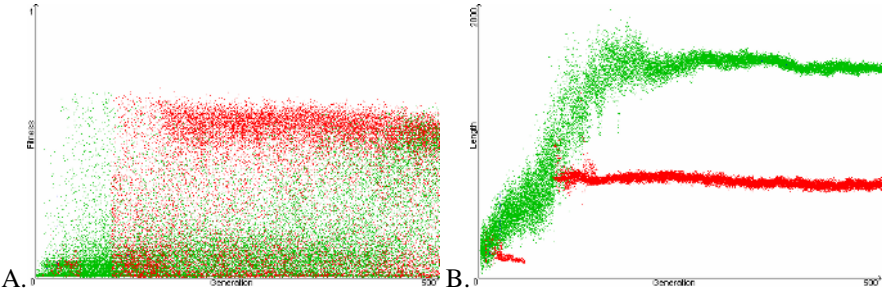


Fig. 4. *(A)* Evolution of population fitness. *(B)* Evolution of population genome length.

The evolution of population fitness and genome length over 500 generations are shown in Fig. 4., as can be seen a high fitness is rapidly obtained within some tens of generations – in addition two different high fitness solutions are obtained as is apparent from the population genome length plots. This is due to the fact that in this application the SVLC algorithm was also coupled with a novel speciation algorithm (Adaptive Breeding Restriction [10]) - notably even in the simple simulation environment used it is possible for two distinct "species" to develop, each following its own compact, relatively converged, course. The behaviour of the two "species" obtained is rather different; the first uses a random wandering approach whilst the second utilises a wall following strategy in order to explore the environment. It is worth noting that fitness is purely determined by the number of pixels collected, no length penalty is used. Despite this the genome lengths of both "species" are relatively stable and are not subject to uncontrolled growth (comparable to "bloat" in Genetic Programming [11]) , instead the genome length has adapted to a suitable length in order to produce controllers that are able to solve the problem.

8 Conclusion

The SVLC algorithm offers a biologically inspired rationale for variable length crossover. It also provides a similarity metric allowing the possibility of using niche formation techniques. In the simple test problem it outperforms current variable

length techniques. The SVLC algorithm also seems to work well in more complex problem domains and is probably applicable to many other variable length problems.

References

1. Goldberg, D., K. Deb, B. Korb.: Messy genetic algorithms: motivation, analysis, and first results. In Complex Systems 3: 493-530 (1989)
2. Fullmer, B., Miikkulainen, R.: Using Marker-Based Genetic Encoding of Neural Networks to evolve Finite-State Behaviour. In Varela, F., Bourgine, P. (eds.), Proceedings of the First European Conference on Artificial Life. MIT Press (1991)
3. Koza, J.: A paradigm for genetically breeding computer programs to solve problems. Technical Report STAN-CS-90-1314, Department of Computer Science, Stanford University (1990)
4. Lee, C., Antonsson, E.: Variable Length Genomes for Evolutionary Algorithms. In Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kauffman, (2000) p. 806
5. Harvey, I.: The SAGA Cross: the mechanics of crossover for variable-length genetic algorithms. In Männer, R., Manderick, B. (eds.), Parallel Problem Solving from Nature 2: 269-278 (1992)
6. Burke, D., De Jong, K., Grefenstette, J., Ramsey, C., Wu, A.: Putting More Genetics into Genetic Algorithms. In Whitley, D. (ed.), Evolutionary Computation 6(4): 387-410 (1998)
7. Kimball, J.: Biology. McGraw Hill Education. (1994)
8. Harvey, I.: Species adaptation genetic algorithms: a basis for a continuing SAGA. In Varela, F., Bourgine, P. (eds.), Proceedings of the First European Conference on Artificial Life, MIT Press, (1992) pp. 346-354
9. Harvey, I.: Artificial Evolution: A Continuing SAGA. In Gomi, T. (ed.), Evolutionary Robotics: From Intelligent Robots to Artificial Life, Proc. of 8th Intl. Symposium on Evolutionary Robotics. Springer-Verlag Lecture Notes in Computer Science LNCS 2217 (2001)
10. Hutt, B.: Evolving Artificial Neural Network Controllers for Robots using Species Based Methods. PhD Thesis, Department of Cybernetics, School of Systems Engineering, The University of Reading (2002)
11. Langdon, W., Poli, R. Fitness Causes Bloat. Proceedings of the Second On-line World Conference on Soft Computing in Engineering Design and Manufacturing (1997).

Valency for Adaptive Homeostatic Agents: Relating Evolution and Learning

Theodoros Damoulas, Ignasi Cos-Aguilera, Gillian M. Hayes, and Tim Taylor

IPAB, School of Informatics, The University of Edinburgh,
Mayfield Road, JCMB-KB, EH9 3JZ Edinburgh, Scotland, UK
{T.Damoulas, I.Cos-Aguilera, G.Hayes, T.Taylor}@ed.ac.uk
http://www.ipab.inf.ed.ac.uk

Abstract. This paper introduces a novel study on the *sense of valency* as a vital process for achieving adaptation in agents through evolution and developmental learning. Unlike previous studies, we hypothesise that behaviour-related information must be underspecified in the genes and that additional mechanisms such as valency modulate final behavioural responses. These processes endow the agent with the ability to adapt to dynamic environments. We have tested this hypothesis with an *ad hoc* designed model, also introduced in this paper. Experiments have been performed in static and dynamic environments to illustrate these effects. The results demonstrate the necessity of valency and of both learning and evolution as complementary processes for adaptation to the environment.

1 Introduction

The relationship between an agent's ability to monitor its internal physiology and its capacity to display adaptation to its environment has received little attention from the adaptive behaviour community. An aspect of this relationship is the sense of *valency*, a mechanism evolved to foster the execution of beneficial actions and to discourage those whose effect is harmful. Agents endowed with this sense exhibit some advantage over others which cannot anticipate the effect of some actions in their decision making. This facilitates the life of an agent in a competitive environment. Formally, we have defined the sense of valency as the *notion of goodness or badness attached by an individual to the feedback from the environment resulting from the execution of a behaviour*. We therefore view valency as a process occurring in a framework of interaction relating perception, internal bodily dynamics and behaviour arbitration. We have implemented these elements in a simulated animat, consisting of an artificial internal physiology [6,5,15], a behaviour repertoire, a selection module and a valency module. The goal of this agent is to survive, ergo to maintain its physiological parameters within their viability zone [2].

Previous work [1,4] hypothesised genes to encode the valency of stimuli and the behavioural responses to stimuli (represented as an evaluation network or as a motivation system, respectively). Both studies use the valency as a feedback loop that assesses and corrects their behavioural patterns. These studies focused

M. Capcarrere et al. (Eds.): ECAL 2005, LNAI 3630, pp. 936–945, 2005.

on the interaction between learning and evolution via the Baldwin effect [3,9], where certain action-related genes dominate and *shield* other genes encoding the valency of related actions. They argue that random mutations allow developmental knowledge to be transferred to the genome, which may deteriorate the valency-related ones. As stated by [1]: *The well-adapted action network apparently shielded the maladapted learning network from the fitness function. With an inborn skill at evading carnivores, the ability to learn the skill is unnecessary.* However, we argue that this may be necessary in a variety of cases, e.g. if the environment constantly changes, it does not seem reasonable to encode volatile information in the genes (this may lead to the extinction of the species). Instead, it seems wiser to genetically encode action-related information in an underspecified manner to be completed via interaction with the environment (via reward driven learning). If as a result of the combination of both processes this information is transferred to the next generation, this would endow the next generation with the necessary knowledge to survive while maintaining the flexibility for a range of variation within its environment.

A model to test this hypothesis is introduced next with three different versions. Section 2 introduces the agent's internal model. Section 3 presents the three approaches examined, their corresponding elements and the results for static and dynamic environments. Finally, Section 4 discusses the results obtained.

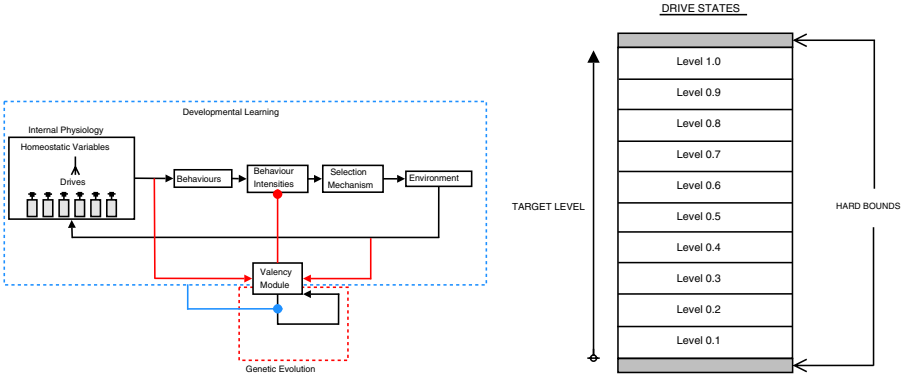
2 Model Architecture

2.1 Internal Agent Structure

The agent's *internal physiology* is a simplified version of the model proposed by Cañamero [5]. In this model the agent's internal resources are represented as *homeostatic variables*. These are characterised by a range of operation and by an optimal value or set point. They exhibit their status of deficit or excess via a set of related *drives* [10], which together with the external stimuli configure the agent's motivational state [19]. For our case we are only interested in the agent's internal interpretation of the effect. Therefore, it is possible to simplify the environment to its minimal expression: the feedback or effect per interaction. This allows us to combine homeostatic variables and drives in a single parameter: *homeostatic drives*. These drives decay according to

$$Level_{(t)} = Level_{(t-1)} \times 0.9 + \sum_{j} < effect\ of\ action >_{t_j} \tag{1}$$

where *level* is the value of a drive and *effect of action* the value of the effect of executing a certain behaviour (an incremental transition of +0.1, 0.0 or -0.1 on the drives). To simplify, the drives are generic (e.g. energy related) since we are mostly concerned with them in terms of their functionality (e.g. decay, discretised states, etc.). Figure 1(b) shows a schematic view of a single drive with its discretised states and the hard bounds.



(a) General Schema (b) Generic Homeostatic Variable

Fig. 1. Left: General Architecture. Right: A typical drive with 10 discretised states from 0.1 to 1.0. The drive has hard bounds below 0.1 and above 1.0 (ad-hoc) to ensure that agents operate within limits.

The *selection mechanism* consists of choosing the action exhibiting the largest valency. As we shall see, the association action-valency is learned during the lifetime of the agent.

2.2 Lifetime Learning

Valency is interpreted by the agent as the relative value of executing a behaviour. This association is learned during lifetime via the *valency module* (cf. center-bottom in Fig. 1(a)) and directly affects the *behaviour intensities* according to the effect that executing a behaviour has on the *internal physiology*.

The learning of the agent is modeled as a 'full' reinforcement learning problem [17]. Every action and state transition of the agent's physiological space is evaluated according to the reward function that is provided by genetic evolution. The learning has been modeled as a Temporal Difference (TD) algorithm [16], since this learns by experience and without *bootstrapping*, i.e., lacking a model for the environment. This should be of advantage in dynamic environments.

The Q-learning algorithm was used with the Q-Values representing the valency of actions and the update rule (2) indirectly associating the effect of an action to a specific valency through the individual reward function.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \, [r_{t+1} + \gamma \, \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]. \qquad (2)$$

2.3 Genetic Evolution

The *valency module* is part of both processes, developmental and genetic. It acts as a link between the two, using genetic information and lifetime experience in order to create an individual *sense of valency*. According to our implementation

the core element of valency is the reward function which is the genetically encoded information. This is *independent* of the behaviours and could be encoded into the genome in a biologically plausible manner.

The *reward function* is evolved by a standard GA [12]; it is either directly encoded in the animat's chromosome or indirectly encoded as the weights of a neural network. In other words, each animat is genetically "defined" to assign a *utility* to each change in its physiological state due to the execution of a behaviour.

The role of genetic evolution and developmental learning in the mechanism of valency, the evolutionary nature (direct or indirect encoding) of the reward function and their effect on adaptation to dynamic environments are, respectively, the issues we have addressed with three different models, introduced in the next section.

3 Experiments and Results

In order to examine the effect of valency in the developmental and genetic processes, this approach has been implemented with direct and indirect encoding of the reward function (Models 1a and 1b), and compared to a model that uses genetic evolution only (Model 2). Models 1a and 1b are used to demonstrate that the instabilities of Darwinian models in dynamic environments [11,14] are due to having *action selection* (as opposed to just the reward function) encoded in the genome. Model 2 is used to examine the necessity of developmental learning in stable and dynamic environments.

Models 1a and 1b test different evolutionary encodings of the reward function. In Model 1a the reward function is directly encoding on the chromosome, whereas in Model 1b the chromosome encodes synaptic weights of a neural network that estimates the reward function. This second encoding method has been extensively used and tested in previous work [4,8,13,14,18].

Finally, we examine the above approaches in both stable and dynamic environments in order to observe their effect on the adaptability of our animats.

3.1 Experimental Setup

The *environment* has been modeled in terms of deterministic reward. Every time the agent performs an action, the environment feedbacks a physiological effect, which is unique for each behaviour. A *static environment* is characterised by relating to each behaviour a unique effect, which is maintained throughout generations (e.g. action 1 has always a -0.1 effect). In contrast, the effect related to each behaviour is inverted every generations for *dynamic environments* (action 1 changes effect from -0.1 to +0.1).

The *Q-Values* represent the value of selecting a specific action in a given state. Q-Values model the valency of actions and qualify an execution as successful or not. Since for every drive we have 10 discrete states and in every state the same action set is available, the Q-Value table for describing every state-action will be

a matrix of dimensions $10 \times (\#Actions\ per\ state \times \#Drives)$. The initialization of the Q-Values has always been performed by setting them to 1 and the update rule 2 converged those values to the individual *valency* of each agent based on its reward function.

The *learning episode* of selecting actions for a number of steps is repeated for at least 10,000 cycles where convergence of the best *possible* behaviour according to the individual reward function is ensured. A *competition* procedure was used to assign the fitness value at each agent at the end of the learning cycle (if applicable). The agent was initialized on a minimum drive level, it was allowed to use its action-selection mechanism (converged Q-values) for a specific number of steps and it was scored according to its overall performance. The target was to reach satiation on its drive(s) and to remain at that level (1.0).

The *metrics* used in our study are the average and maximum fitness progressions through generations of animats. The maximum fitness score each time (10 for single drive and 20 for double drive cases) indicates a perfect performance over the competition cycle and a successfully evolved/developed sense of valency.

3.2 Learning and Evolution with Indirect Encoding of Action Selection

As has been shown previously [1,11,14], direct encoding of action selection leads to animats that are behaviouraly predisposed. Consequently, their fitness progressions in dynamic environments suffer from relative instabilities. To overcome these limitations, our **Model 1a (RL & GA)** was investigated (Fig. 2), where the genome is not directly encoding action-selection. Instead it carries information (reward for state transitions) used to build the *behaviour-selection* mechanism via developmental learning.

An alternative version of the above implementation, **Model 1b (RL & GA with NN**), which uses a neural network for the provision of the reward function (Fig. 3), was also examined. The genome is still *indirectly* encoding action selection.

3.3 Strictly Evolutionary Approach

The final model (Model 2) was used to test the necessity of lifetime learning as a complementary process to genetic evolution. **Model 2 (Q-Evolution)** is strictly evolutionary (Fig. 4).

3.4 Learning and Evolution, or Evolution Alone?

Static Environment. The *base case* considered first is that of a static environment with an animat endowed with a single drive. As seen in Fig. 5, every approach manages to produce ideal solutions, i.e., animats with a sense of valency are able to guide selection toward the satiation of the drive. The results confirm our hypothesis that a combined framework of learning and evolution through the valency module performs better than those lacking it. However, the strictly evolutionary approach (Q-Evolution model) still manages to achieve,

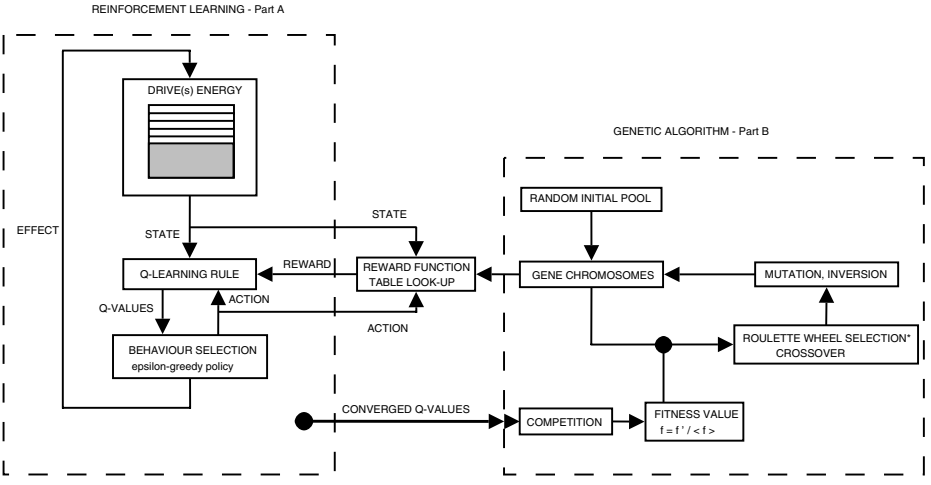


Fig. 2. Model 1a (RL & GA) of combined Learning and Evolution. The chromosome encodes the reward function of each agent (magnitudes in the range 0-10)and the Q-learning update rule is used to create the action selection mechanism through lifetime. Step-size parameter α=0.1 and ϵ=0.1

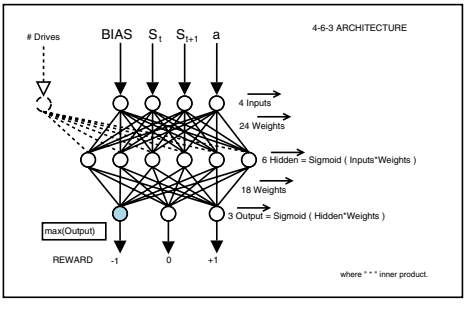


Fig. 3. The Neural Network architecture used in Model 1b (RL & GA with NN). The input is the states, the bias, and the action whereas the output is the magnitude of reward. In this case a simple set of reward was used with +1, 0 or -1 possible values. The bias is set to 1 and the network operates with a standard sigmoid function. In the case of more then one drives an additional node inputs that information.

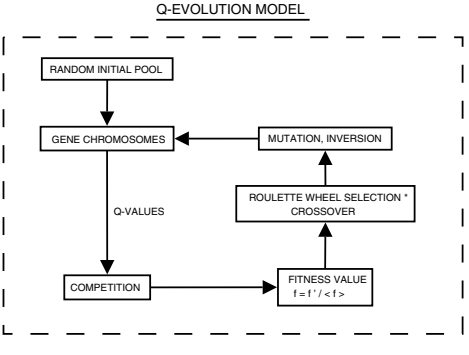
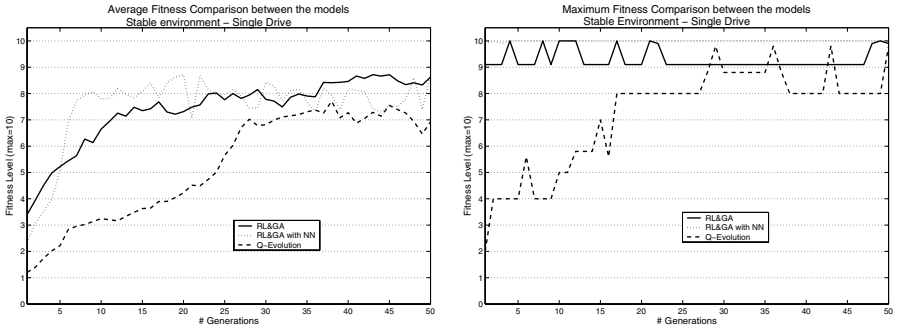
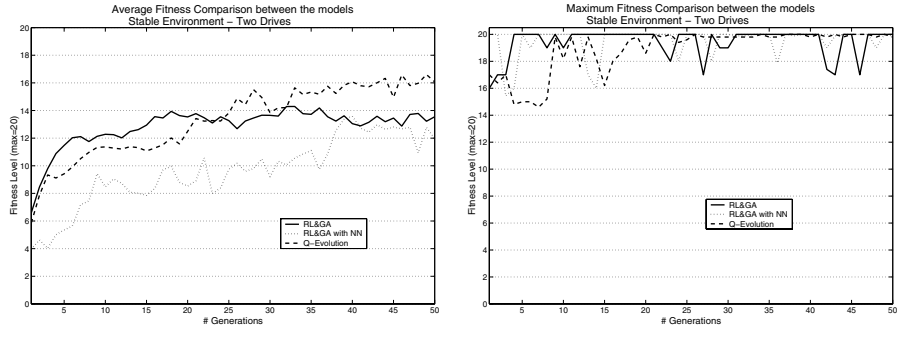


Fig. 4. Model 2 (Q-Evolution), implementing only standard Evolutionary techniques without a Learning cycle. The valencies of actions (Q-Values) are directly evolved instead of the reward function and hence the genome of agents encodes information that is directly connected to the action selection mechanism. The model is operating without a valency module.



(a) Average (b) Maximum

Fig. 5. Average and Maximum Fitness results for the three models on a stable environment where the animats have a single drive. The fitness function requires optimal behaviour selection in order to achieve maximum status. Notice how the models utilizing developmental learning achieve higher fitness levels in a few number of generations.



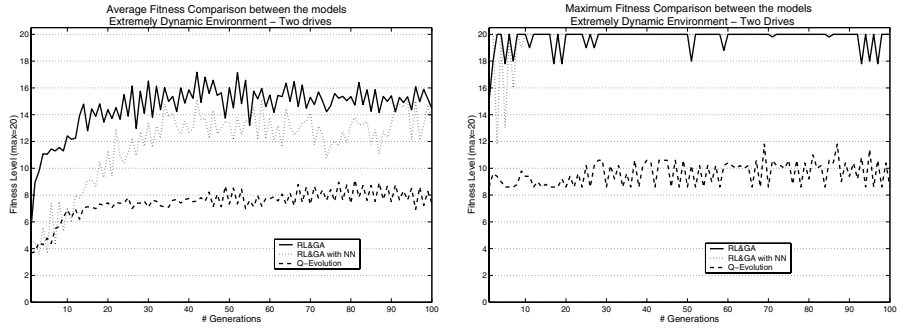
(a) Average (b) Maximum

Fig. 6. Average and Maximum Fitness results for the three models on a stable environment where the animats are utilizing two drives. The results are for a "loose" fitness function that allows suboptimal behaviour selection.

in certain occasions, maximum fitness and to increase the average of the population. The approach that directly evolves the reward function (RL & GA) achieves a higher average fitness but is less stable in the maximum fitness development compared with the alternative evolution of synaptic weights (RL & GA with NN).

The *double drive case* in a stable environment increases the difficulty of the task and explores the capabilities of all the approaches. The results in Fig. 6 compare the models on a "loose" fitness function (excess competition steps) that allows for suboptimal behaviour selection (the animat can achieve maximum fitness without selecting always the best possible action). For a fitness function requiring optimal behaviour selection (that is, always to choose the best behaviour), the strictly evolutionary approach fails to produce a perfect solution even after 50,000 generations [7].

Dynamic Environments. The effect of the *dynamic* environment on the adaptability of the animats is shown in Fig. 7. The extreme dynamic case is considered where the effect of actions is changing at every generation. Under these circumstances the models implementing a combined framework of learning and evolution via an indirect encoding of action-selection, manage to produce animats able to adapt to the environment overcoming the initial fluctuations on the maximum fitness of the population.



(a) Average (b) Maximum

Fig. 7. Average and Maximum Fitness results for the three models on a dynamic environment where the animats are utilizing two drives. Every 1 generation the environment changes, causing the effect of actions to be inverted. Only the models implementing both developmental and genetic approaches are adaptable to the changes and able to achieve consecutive maximum fitness solutions. The Q-Evolution model is unstable.

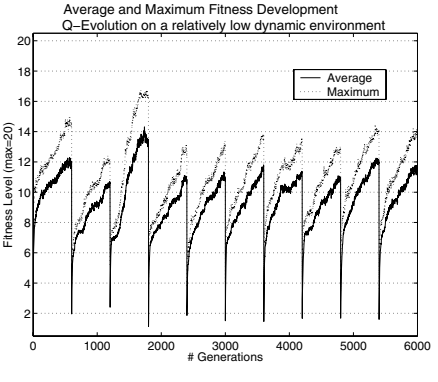


Fig. 8. The Q-Evolution model implementing an evolutionary approach without developmental learning fails to adapt even to a relative low-dynamic environment that changes every 600 generations. The limitation of the model is due to the lack of a complete valency module. Whenever the environment suffers a change there is a sudden drop on both the average and maximum fitness level of the population.

In contrast, the Q-Evolution model, which implements a strictly evolutionary approach, is unable to adapt to the dynamic environment as it is shown by the low-value and fluctuating average and maximum fitness developments. Even in a dramatically less severe environment, where the changes occur every 600 generations (Fig. 8), evolution alone is unable to follow the changes of the

environment and both the average and maximum fitness of the population have a sudden drop at the instance of the change.

3.5 Direct or Indirect Encoding of Action Selection?

Contrary to the results of [11,14], the average fitness progression of the combined learning and evolution approach does not suffer from large oscillations every time the environment changes. This is due to the fact that action selection is underspecified in the genes and hence the animats do not have to unlearn *and* relearn the right behaviour. They just have to learn it during their lifetime. This demonstrates and proves our hypothesis that underspecified encoding of action selection, in a combined framework of developmental learning and genetic evolution, endows animats with a further adaptive skill that facilitates their survival.

In contrast, animats with an "inborn" skill for selecting and executing a behaviour have to re-learn it at every change of the feedback from the environment. This is a dramatic disadvantage, leading to the animats' extinction when the genetically encoded behaviour becomes a deadly option.

4 Discussion and Conclusion

In the present study we have examined the role of valency as a process relating developmental learning and genetic evolution to assist adaptation. We implemented two different approaches, one that is strictly evolutionary and one that makes use of both developmental and evolutionary mechanisms in order to compare and draw conclusions on the nature of valency. Furthermore, we have tested their performance on both stable and dynamic environments in order to investigate their adaptability.

It has been demonstrated that in both stable and dynamic environments a combined framework of learning and evolution performs better, since agents achieve higher fitness in fewer generations. In the case of an animat equipped with two drives, or in a dynamic environment, evolution alone fails to find a perfect solution, implying that a valency mechanism is necessary if the animats are to adapt at all. Furthermore, we have shown that action selection has to be underspecified in the genome for the sake of adaptation. Instead of directly encoding action selection (as in [1,14,11]), the genes should *indirectly* encode that information in order to avoid becoming predisposed toward the execution of a behaviour that could later become harmful.

References

1. Ackley, D. and Littman, M.: Interactions between learning and evolution. In C. Langton, C. Taylor, D. Farmer, and S. Rasmussen, editors, Proceedings of the Second Conference on Artificial Life. California: Addison-Wesley, 1991.
2. Ashby, W.R.: Design for a Brain: The Origin of Adaptive Behaviour. Chapman & Hall, London, 1965.

3. Baldwin, J.M: A new factor in evolution. The American Naturalist, 30 (June 1896):441-451, 536-553, 1896.
4. Batali, J. and Grundy, W.N: Modelling the evolution of motivation. Evolutionary Computation, 4(3):235-270, 1996.
5. Cañamero, D.: A hormonal model of emotions for behavior control. In VUB AIMemo 97U06, Free University of Brussels, Belgium. Presented as poster at the Fourth European Conference on Artificial Life. ECAL 97, Brighton, UK, July 28-3, 1997.
6. Cos-Aguilera, I., Cañamero, D. and Hayes, G.: Motivation-driven learning of object affordances: First experiments using a simulated khepera robot. In Frank Detje, Dietrich Dörner, and Harald Schaub, editors, The Logic of Cognitive Systems. Proceedings of the Fifth International Conference on Cognitive Modelling, ICCM, pages 57-62. Universitäts-Verlag Bamberg, April 2003.
7. Damoulas, T.: Evolving a sense of Valency. Masters thesis, School of Informatics, Edinburgh University, 2004.
8. Harvey, I.: Is there another new factor in evolution? Evolutionary Computa- tion, 4(3):313-329, 1996.
9. Hinton, G.E. and Nowlan, S.J.: How learning can guide evolution. Complex Systems, 1:495-502, 1987.
10. Hull, C.: Principles of Behaviour: an Introduction to Behaviour Theory. D. Appleton-Century Company, Inc., 1943.
11. Mclean, C.B.: Design, evaluation and comparison of evolution and reinforcement learning models. Masters thesis, Department of Computer Science, Rhodes University, 2001.
12. Mitchell, M.: An Introduction To Genetic Algorithms. A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, 1998.
13. Nolfi, S.: Learning and evolution in neural networks. Adaptive Behavior, (3) 1:5-28, 1994.
14. Sasaki, T. and Tokoro, M.: Adaptation toward changing environments: Why darwinian in nature? In P. Husband and I. Harvey, editors, Proceedings of the Fourth European Conference on Artificial Life, pages 378–387. MIT Press. 1997.
15. Spier, E. and McFarland, D.: Possibly optimal decision-making under self-sufficiency and autonomy. Journal of Theoretical Biology, (189):317-331, 1997.
16. Sutton, R.S.: Learning to predict by the method of temporal difference. Machine Learning, 3(1):9-44, 1988.
17. Sutton, R.S. and Barto, A.G.: Reinforcement Learning. MIT Press, 1998.
18. Suzuki, R. and Arita, T.: The baldwin effect revisited: Three steps characterized by the quantitative evolution of phenotypic plasticity. Proceedings of the Seventh European Conference on Artificial Life (ECAL2003), pages 395-404, 2003.
19. Toates, F. and Jensen, P.: Ethological and psychological models of motivation - towards a synthesis. In Jean-Arcady Meyer and Stewart W. Wilson, editors, Proceedings of the First International Converence on Simulation of Adaptive Behaviour, A Bradford Book., pages 194-205. The MIT Press, 1990.

Author Index