George Kampis
István Karsai
Eörs Szathmáry (Eds.)

Advances in Artificial Life

Darwin Meets von Neumann

LNAI 5777

10th European Conference, ECAL 2009
Budapest, Hungary, September 2009
Revised Selected Papers, Part I

Springer

Lecture Notes in Artificial Intelligence 5777

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

George Kampis István Karsai
Eörs Szathmáry (Eds.)

Advances in Artificial Life

Darwin Meets von Neumann

10th European Conference, ECAL 2009
Budapest, Hungary, September 13-16, 2009
Revised Selected Papers, Part I

Springer

Volume Editors

George Kampis
Eötvös University
Department of History and Philosophy of Science
Pázmány P.s. 1/C, 1117 Budapest, Hungary
E-mail: gk@hps.elte.hu

István Karsai
East Tennessee State Unviversity
Department of Biological Sciences
Johnson City, TN 37614-1710 USA
E-mail: karsai@etsu.edu

Eörs Szathmáry
Collegium Budapest, Szentháromság u.2
1014 Budapest, Hungary
E-mail: szathmary@colbud.hu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-21282-6 e-ISBN 978-3-642-21283-3
DOI 10.1007/978-3-642-21283-3

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011927893

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

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Artificial Life, unlike artifical intelligence, had humble beginnings. In the case of the latter, when the word itself was born, the first breathtaking results were already out, such as The Logic Theorist, a computer program developed by Allen Newell, Herbert Simon and Cliff Shaw in 1955–56. In artifical life, for a long while, amibition seems to have dominated over the results, and this was certainly true for the first, formative years. It was a bit unclear what *exactly* Artificial Life is. As not uncommon in science, the first definition was attempted in the negative form, just like when psychology (the study of the mental) was first defined as "anything not physics" (meaning, not natural science) in the nineteenth century. A tempting early definion of Artificial Life was one that distinguished it from theoretical and mathematical biology, modeling, evolution theory, and all the rest of what consituted "an old kind of science" about life. This was not without grounds, perhaps, and the parallel with artificial intelligence comes up again. Artificial intelligence was conceptually based on "machine functionalism," the philosophical idea that all operations, such as the mental operations of humans, are to be captured as "mere functions," or, in other words, as independent of their actual physical realizations. Functionalism has put computers and algorithms in the focus of interest in all dealings with human intelligence, and artificial intelligence was a computerized approach to the mind that was designed to capture human mental operations in the functional sense. Now it was simply the case that the *functionalism of life* was not yet born, and Artificial Life looked like the candidate that was needed for exactly that—to discover how computers can be used to uncover the secrets of life. There were cellular automata, that John von Neumann discovered back around 1950, that are capable of self-reproduction. Perhaps life was just a step away. This and a new fascination with functionalism in Artificial Life put computer scientists (who could technically master cellular automata and computer math) into a central position—in Artificial Life as it could be.

But Artificial Life became different. Incidentally, the slogan "life as it could be" was coined as a motto for Artificial Life, but now the same conditionals apply to Artificial Life itself. The reason is that functionalism turned out to be just one part of the story.

There is a well-known and much used (maybe over-used) phrase in biology, called "Dobzhansky's dictum," which says that "nothing in biology makes sense except in the light of evolution." Evolution is, as rightly captured in the dictum, central to the understanding of all things alive, and hence it is, and this had to be discovered, central to the studies of Artificial Life as well. And soon it also turned out that evolution cannot be readily reduced to algorithmic problems, or and least not in that pure, detached sense as it was attempted in functionalism. Evolution is *complex* in a sense recently acknowledged in the sciences of complexity: there is no single principle, no simple set of rules, and no transparency. Instead,

evolution is a combination of many heterogeneous, and sometimes contingent factors, many of which have to do with the complex ways of existence of organisms: their body, their interaction, their development, their geo-spatial relations, their temporal history, and so on. This brought biology and biologists back into the equation, and Artificial Life has greatly benefited from that. Evo-devo (the interplay between evolutionary and developmental biology), evolutionary robotics, or systems biology are examples of fields where mathematical and algorithmic thinking combined with "wet" reality started to produce new and fascinating results. (Those who kept an eye on cognitive science and artificial intelligence found that over all those years a similar process has taken place there as well. *Embodiment*, or situated physical realization, has permeated and changed these fields to the same extent, or even more, as it did Artificial Life).

Today, as a result of these processes, Artificial Life is a prolific field that combines the best of computer science with the best of theoretical biology, mathematical modeling, and simulation. One way to express this is to say "Darwin meets von Neumann" at last—where "real" biology and "pure" function are no longer seen as contradicting, or even complementary. Rather, they permeate and fertilize each other in a number of fascinating ways.

ECAL 2009 was the 10th Europan Conference of Artificial Life, which means 20 years of history. It was an occasion to celebrate the 20 years of development of the field and also the new symbiosis referred to in the title. Famously, 2009 was also "Darwin year," celebrating his 200th birthday and the 150 years of the *Origin of Species*. Thus it was highly appropriate to dedicate the meeting to Darwin—and von Neumann together. Five keynote lectures were delivered by eminent invited speakers, in the order of appearance they were: Peter Hammerstein (Humboldt University, Berlin), Hod Lipson (Cornell), Nick Barton (FRS, Edinburgh), Richard Watson (Southampton) and Eva Jablonka (Tel-Aviv University)—their choice reflected the spirit of convergence alluded to above.

The conference featured 161 submissions, out of which 54 were accepted as talks and 87 as posters (making up a total of 141 presentations). Adopting the recent practice of many science meetings, submissions could be based either on full papers or extended abstracts. The meeting was organized in a single track over three days, with parallel (whole-day) poster sections, to give best visibility to everyone's work. We decided to publish all papers of the accepted presentations, not making any difference between posters and talks. This resulted from different factors: many excellent submissions had to be put into the poster section to keep reasonable time limits for the talks, and often this included second or third papers of some of the key speakers. Poster or talk was therefore not necessarily a quality issue. But also, we decided to publish all poster papers because we wanted to show the heterogeneity and the full spectrum of activities in the field, in order to provide an authentic overview. The result is this volume in 2 parts, containing 116 full papers.

The conference was sponsored by the Hungarian Academy of Science in different ways, one of them the special rates we enjoyed when using the wonderful historical building of the Academy in the very center of Budapest, just across

the castle and the Chain Bridge. It is a building with a unique historical atmosphere and one that has seen many major scientific events. The conference talks were held in the Great Lecture Hall, which added to the impression that Artificial Life—and ECAL—are coming of age. The other sponsor was Aitia International, Inc., whose support is gratefully acknowledged. Aitia is the maker of MEME, or Model Exploration ModulE, a platform for DoE (design of experiments) and parameter sweeping, running on a cloud (https://modelexploration.aitia.ai/).

The publication process experienced several unexpected difficulties and delays. The proceedings could never have been published without a final push by Mark Jelasity, of Szeged University, a member of the Local Organizing Committee. It was his support and his offer for a hands-on contribution and equally his expertise of LaTeX and prior experience with Springer LNCS publications that made the essential difference that helped cross the line. Mark was offered but declined to be an Editor, lacking a scientific contribution to this conference and bearing a responsibility for the selection process, and this is a decision we had to respect. Nevertheless, here is a "big thank you," Mark. Several other people provided important help in the production of the volume, of whom Balazs Balint (Collegium Budapest) and Chrisantha Fernando (Sussex) need special mention. We thank the TenSi Congress Ltd. for the seamless technical organization of the meeting. In the evaluation phase, important help was given by several members of the Program Committee and also by additional reviewers, listed separately, whose contribution is highly appreciated. The conference and the proceedings have been the work of several people, and we thank all of them for making it happen. Finally, we thank Anna Kramer of Springer for her support and patience.

February 2011

George Kampis
István Karsai
Eörs Szathmáry (Editors)

Organization

Organizing Committee

George Kampis and Eörs Szathmáry (Chairs)
Chrisantha Fernando, Márk Jelasity, Ferenc Jordán, András Lőrincz, and István Scheuring

Program Committee

Wolfgang Banzhaf
Xabier Barandiaran
Zoltan Barta
Mark Bedau
Randall Beer
Seth Bullock
Philippe Capdepuy
Peter Cariani
Andy Clark
Tamás Czárán
Ezequiel Di Paolo
Dario Floreano
Inman Harvey
Takashi Ikegami
Istvan Karsai
Jozef Kelemen
Simon Kirby
Doron Lancet
Robert Lowe
John McCaskill
Bruce MacLennan
Federico Moran
Alvaro Moreno
Chrystopher Nehaniv
Stefano Nolfi
Charles Ofria
Alexandra Penn
Daniel Polani
Luis Rocha
Matthias Scheutz
Thomas Schmickl
Peter Stadler
Elio Tuci
Andreas Wegner
Franjo Weissing
Larry Yaeger
Klaus-Peter Zauner

Additional Reviewers

Rose Canino-Koning
Arthur Covert
Tomassino Ferrauto
Sherri Goings
Heather Goldsby
Laura Grabowski
David Knoester
Anuraag Pakanati
Bess Walker
Luis Zaman

Table of Contents – Part I

Evolutionary Developmental Biology and Hardware

Evolutionary Robotics

Protocells and Prebiotic Chemistry

Systems Biology, Artificial Chemistry and Neuroscience

The Evolution of Cooperation

Table of Contents – Part II

Group Selection

Ecosystems and Evolution

Algorithms and Evolutionary Computation

Philosophy and Arts

Optimization, Action, and Agent Connectivity

Swarm Intelligence

Self-organizing Biologically Inspired Configurable Circuits

André Stauffer and Joël Rossier

Ecole polytechnique fédérale de Lausanne (EPFL), Logic Systems Laboratory
CH-1015 Lausanne, Switzerland
Tel.: (+41 21)693 26 52
andre.stauffer@epfl.ch

Abstract. Inspired by the basic processes of molecular biology, our previous studies resulted in defining self-organizing mechanisms made up of simple processes. The goal of our paper is to introduce a configurable molecule able to implement these mechanisms as well as their underlying processes. The hardware description of the molecule leads to the simulation of an arithmetic and logic unit designed as a one-dimensional organism dedicated to bit slice processors.

1 Introduction

Borrowing the structural principles from living organisms, we have already shown how to grow cellular systems thanks to an algorithm for cellular division [2]. These cellular systems are endowed with self-organizing properties like configuration, cicatrization, and regeneration.

In a previous work [3], the configuration mechanisms (structural and functional growth), the cicatrization mechanism (cellular self-repair), and the regeneration mechanism (organismic self-repair) were already devised as the result of simple processes like growth, load, repair, reset, and kill. The goal of this paper is to devise a configurable molecule able to perform these self-organizing mechanisms and their underlying processes in order to design biologically inspired circuits.

Starting with the detailed architecture of the configurable molecule, Section 2 will point out how bit slice processors represent a perfect example of such biologically inspired circuits. We introduce then digital simulations, based on the hardware description of the configurable molecule (Section 3), in order to perform the self-organizing mechanisms on a minimal slice. The simulations of Section 4 apply these mechanisms in the building and maintaining of an arithmetic and logic unit made up of three slices. A brief conclusion (Section 5) summarizes our paper and opens new research avenues.

2 Configurable Molecule

Our biologically inspired circuits are designed as two-dimensional arrays of configurable molecules. Each molecule of these arrays is made up of a configuration layer and an application layer.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 1–9, 2011.

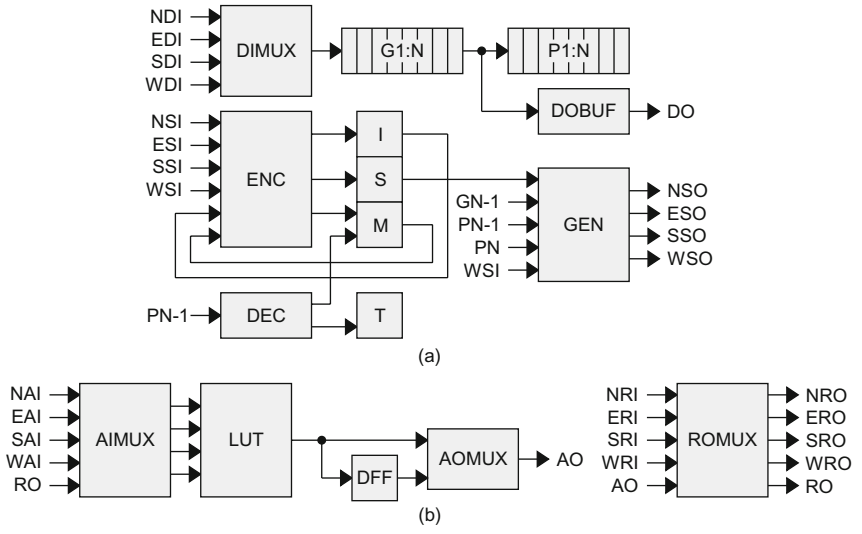


Fig. 1. Detailed architecture of the configurable molecule. (a) Configuration layer. (b) Application layer.

The *configuration layer*, which implements the self-organizing mechanisms and their constituting processes [4], results from the interconnection of the ten following resources (Fig. 1a): (1) an input multiplexer DIMUX, selecting one out of the four northward NDI, eastward EDI, southward SDI or westward WDI configuration input data, (2) a 2N-level stack organized as N genotypic registers G1 to GN (for mobile configuration data), and N phenotypic registers P1 to PN (for fixed configuration data), (3) an output buffer DOBUF producing the configuration output data DO, (4) an encoder ENC for the northward NSI, eastward ESI, southward SSI, and westward WSI input signals, (5) a decoder DEC defining the mode and the type of the molecule, (6) a register I for the memorization of the input selection, (7) a register S for the transmission of the signals, (8) a register M for the molecular modes, (9) a register T for the molecular types, and (10) a generator GEN producing the northward NSO, eastward ESO, southward SSO, and westward WSO output signals.

The *application layer*, which implements the logic design of the application under development as well as its routing connections between neighboring and distant molecules, results from the interconnection of the five following resources (Fig. 1b): (1) an input multiplexer AIMUX, selecting four inputs out of the four northward NAI, eastward EAI, southward SAI, westward WAI application data, and the routing data RO, (2) a 16-bit look-up table LUT, (3) a D-type flip-flop DFF for the realization of sequential circuits, (4) an output multiplexer AOMUX selecting the combinational or the sequential data as application output AO, and (5) an output multiplexer ROMUX selecting the five outputs NRO, ERO, SRO, WRO, and RO out of the four northward NRI, eastward ERI, southward SRI, westward WRI routing input data, and the application output data AO.

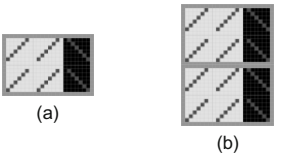


Fig. 2. Bit slice processor. (a) Minimal cell made up of six molecules. (b) Minimal organism made up of two cells.

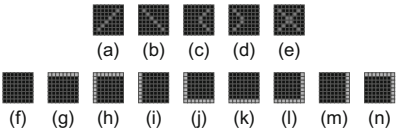


Fig. 3. Molecular modes. (a) Living. (b) Spare. (c) Faulty. (d) Repair. (e) Dead. Molecular types. (f) Internal. (g) Top. (h) Top-left. (i) Left. (j) Bottom-left. (k) Bottom. (l) Bottom-right. (m) Right. (n) Top-right.

Bit slice processors represent a perfect example of application for biologically inspired circuits. They are made up of identical slices and can be seen as multicellular organisms made up of identical cells. Each slice computes at least one data bit and corresponds to a cell made up of configurable molecules. The minimal cell consists of two rows of three molecules with two columns of living molecules dedicated to the slice specifications to the left and one column of spare molecules to the right (Fig. 2a). The minimal multicellular organism is made up of two identical cells and represents a bit slice processor computing at least two data bits (Fig. 2b). The corresponding molecular modes and molecular types are shown in Fig. 3.

3 Self-organizing Mechanisms

3.1 Configuration Test

Performed on a given array of molecules, the purpose of the *configuration test mechanism* is to kill all the columns of molecules having at least a faulty one among them. A molecule is faulty when the shift operation performed by its configuration registers, the genotypic registers G1 to GN as well as the phenotypic registers P1 to PN (Fig. 1a), presents an incorrect behavior. The configuration test mechanism is made up of a growth process followed by a kill process for each detected error and finally a reset process.

Executed using *growth signals* and according to a predefined test configuration string, the *growth process* starts the building of tree shaped datapaths all over the array until a faulty molecule is detected (Fig. 4a-b). The building of the datapaths resumes after the death of the second left column of molecules (Fig. 4e-g).

As soon as a malfunction of its configuration registers occurs, the molecule enters the dead mode and sends *kill signals* northward and southward in order to

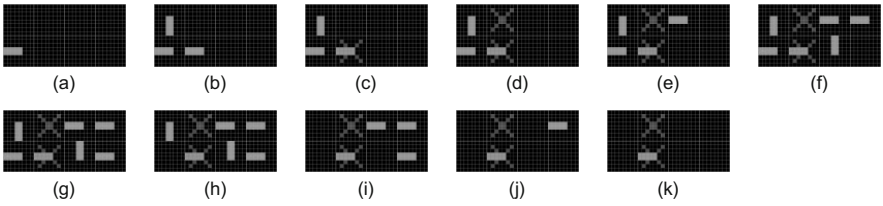


Fig. 4. Configuration test mechanism. (a-b,e-g) Growth process. (c-d) Kill process. (h-k) Reset process.

trigger the death of the whole column of molecules. Fig. 4c-d illustrates the *kill process* involved in the configuration test mechanism by the incorrect behavior of the second lower left molecule.

At the end of the growth process, all the molecules of any column having at least a faulty one are dead. Performed on the array resulting from the malfunction of the second lower left molecule, the *reset process* starts from the lower left molecule and propagates *reset signals* eastward and northward in order to destroy the datapaths builded among the healthy molecules (Fig. 4h-k). This tissue, comprising now one column of dead molecules, is ready for being configured.

3.2 Structural Configuration

The goal of the *structural configuration mechanism* is to define the boundaries of the cell as well as the living mode or spare mode of its constituting molecules. This mechanism is made up of a structural growth process followed by a load process.

The *growth process* starts when an external *growth signal* is applied to the lower left molecule of the cell. This molecule selects the eastward data input (Fig. 5a) and according to the *structural configuration data* or *structural genome*, each molecule of the cell generates then successively an internal *growth signal* and selects an input in order to create a datapath among the molecules of the cell (Fig. 5a-f). When the connection path between the molecules closes (Fig. 5g), the lower left molecule delivers a *close signal* to the nearest left neighbor cell. The structural configuration data is now moving around the datapath and ready to be transmitted to neighboring cells.

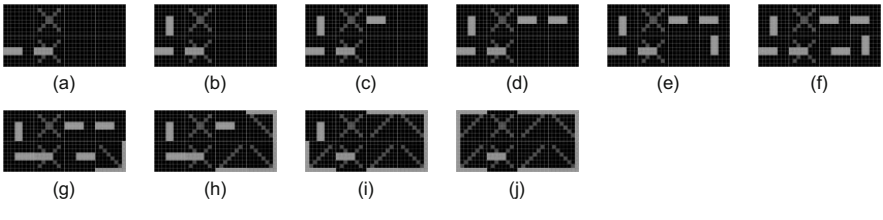


Fig. 5. Structural configuration mechanism. (a-f) Growth process. (g-j) Load process.

The *load process* is triggered by the *close signal* applied to the lower right molecule of the cell. A *load signal* propagates then westward and northward through the cell (Fig. 5g-j) and each of its molecules acquire a *molecular mode* and a *molecular type* (Fig. 3). We finally obtain an homogeneous array of molecules defining both the boundaries of the cell and the position of its *living mode* and *spare mode* molecules (Fig. 5j). This array is ready for being configured by the functional configuration data.

3.3 Functional Configuration

The goal of the *functional configuration mechanism* is to store in the homogeneous array, which already contains structural data (Fig. 5j), the functional data needed by the specifications of the current application. This mechanism is a *functional growth process*, performed only on the molecules in the living mode while the molecules in the spare mode are simply bypassed. It starts with an external *growth signal* applied to the lower left living molecule. According to the *functional configuration data* or *functional genome*, the living molecules then successively generate an internal *growth signal*, select an input, and create a path among the living molecules of the cell (Fig. 6a-e). The functional configuration data is now moving around the datapath and ready to be transmitted to neighboring cells.

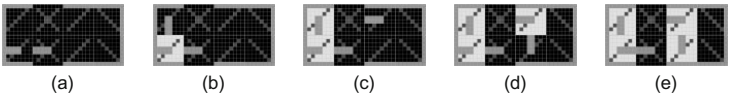


Fig. 6. Functional configuration mechanism. (a-e) Growth process.

3.4 Control Test

In order to correct deterioration that could affect the mobile functional configuration data, the *control test mechanism* is made up of a reset process followed by a functional growth process. This mechanism is performed when the data moving around the cell are different of the ones moving around its western and southern neighbors. The comparison is realized by the lower left molecule of the cell.

When a difference occur, the lower left molecule starts the *reset process* and propagates *reset signals* eastward and northward in order to destroy the datapath build among the healthy molecules of the array. Fig. 7a-c displays the reset process applied to previously configured array.

Performed according to the functional configuration data corresponding to the specifications of the current application, the *growth process* rebuilds the datapath among the living molecules of the cell. This process starting from the lower left molecule is shown in Fig. 7d-h. It renews the mobile data moving around the cell as well as the fixed data of its molecules.

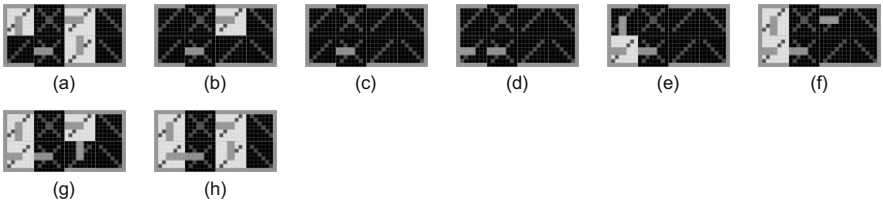


Fig. 7. Control test mechanism. (a-c) Reset process. (d-h) Growth process.

3.5 Processing Test

The *processing test mechanisms* are introduced in order to repair a cell having molecules that present an incorrect behavior at the functional application level. Depending on the number of faulty molecules in a same row between two spare columns, the processing test mechanism results respectively in a cicatrization mechanism or in a regeneration mechanism.

In order to introduce error detection at the application layer level, the architecture of this level has to be doubled. The detection is then made by comparing the application data *AO1* and *AO2* of the their two output multiplexers AOMUX (Fig. 1b).

Starting with the normal behavior of Fig. 6e, we suppose that the upper left molecule becomes suddenly faulty and triggers a *cicatrization mechanism*. This mechanism is made up of a *repair process* involving eastward propagating *repair signals* (Fig. 8a-c) followed by a *reset process*, starting from the upper right molecule, performed with westward and southward propagating internal *reset signals* (Fig. 8d-f). This array, comprising now one molecule in the faulty mode and two molecules in the repair mode, is ready for being reconfigured by the functional configuration data. This implies a *growth process* bypassing the faulty molecule (Fig. 8g-k).

Our cell comprises a single spare molecule per row and tolerates therefore only one faulty molecule in each row. A second faulty molecule in the same row will activate a *regeneration mechanism* and cause the death of the whole cell. Starting with the normal behavior of the cicatrized cell (Fig. 8k), a new molecule, the upper right one, detects an error. Being previously already in the

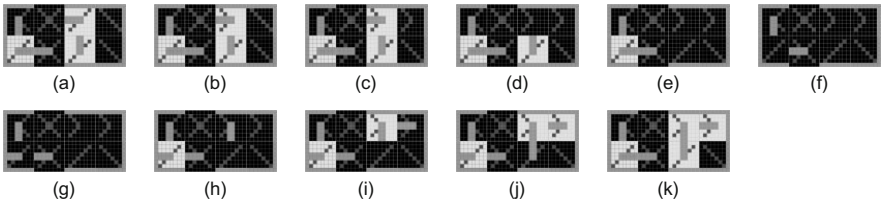


Fig. 8. Cicatrization mechanism. (a-c) Repair process. (d-f) Reset process. (g-k) Growth process.

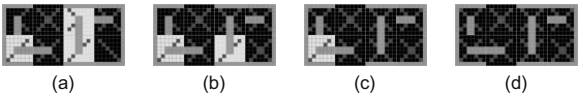


Fig. 9. Regeneration mechanism. (a-d) Kill process.

repair mode, this molecule enters the lethal dead mode and triggers *kill signals* which propagate northward, westward and southward (Fig. 9a-d). Finally, all the molecules of the array are dead as well as the entire cell.

4 Arithmetic and Logic Unit Application

Even if the final goal is the self-organization of bit slice processors, we will use a simplified application example in order to illustrate its basic mechanisms. The circuit that perform arithmetic and logic operations on two 3-bit data A and B can be considered as a one-dimensional artificial organism composed of three identical cells. Each cell is made up of six application specific molecules (Fig. 10): a C molecule computing the carry output, a G molecule computing the generate carry signal, a P molecule computing the propagate carry signal, an R molecule computing the result, an O molecule recovering the result performed by the living organism, and a D molecule generating a deactivation signal in order to bypass the cells of the neighboring spare organism to the right.

In order to build the multicellular organism of Fig. 11a, the configuration test mechanism is performed on an array of five by six molecules and leads to an entire column of dead molecules. The structural configuration mechanism and the functional configuration mechanism are then applied at the cellular level. Starting with the structural and functional configuration data of the basic cell, these mechanisms generate successively the three identical cells of the minimal organism. In this implementation, each individual cell of the organism presents one column of dead molecules and one column of spare molecules.

The cicatrization mechanism (or cellular self-repair) results from the introduction of the column of spare molecule (Fig. 11a), defined by the structural configuration of the basic cell, and the detection of faulty molecules. Thanks to this mechanism, the faulty molecule of the lower cell (Fig. 11b) is deactivated,

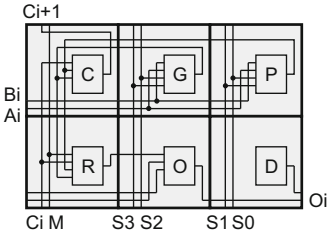


Fig. 10. Basic cell of the 3-bit data A and B arithmetic and logic unit

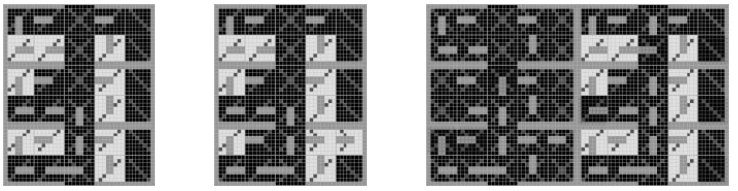


Fig. 11. Arithmetic and logic unit. (a) One-dimensional organism made up of three cells. (b) Graphical distortion resulting from the cicatrization mechanism applied to the lower cell. (c) Scar resulting from the regeneration mechanism applied to the left organism.

isolated from the network, and replaced by the nearest right molecule, which will itself be replaced by the nearest right molecule, and so on until a spare molecule is reached. The functional reconfiguration mechanism takes then place in order to regenerate the organism. As shown in Fig. 11b, the regenerated organism presents some graphical distortion.

Each individual cell of the organism having a single spare column (Fig. 11a), this implementation allows at most one faulty molecule per row. When two of them are detected in a given row, the regeneration mechanism (or organismic self-repair) takes place and all the cells of the organism are considered faulty and are deactivated. The functions of the faulty cells are thus shifted to the spare cells to the right. Obviously, this process requires at least one spare organism to the right. As shown in Fig. 11c, the repair of the faulty organism needs the spare organism to the right and leaves a scar in the implementation.

5 Conclusion

This paper is a contribution to the embryonic project [1] which is dedicated to the building of biologically inspired circuits in silicon. It supplies the detailed architecture of a configurable molecule made up of a configurable layer and an application layer.

Using the VHDL description language, we have realized the hardware implementation of the configuration layer and the application layer of the configurable molecule. The hardware simulations of the minimal slice and of the arithmetic and logic unit presented in the paper are performed on arrays of such molecules.

The configurable molecule, based on the VHDL descriptions of its layers, will be implemented in the *ubichip* [5], a programmable circuit that draws inspiration from the multi-cellular structure of complex biological organisms.

References

1. Canham, R., Tyrrell, A.M.: An embryonic array with improved efficiency and fault tolerance. In: Lohn, J., et al. (eds.) Proceedings of the NASA/DoD Conference on Evolvable Hardware (EH 2003), pp. 265–272. IEEE Computer Society, Los Alamitos (2003)

2. Mange, D., Stauffer, A., Petraglio, E., Tempesti, G.: Self-replicating loop with universal construction. Physica D 191(1-2), 178–192 (2004)
3. Stauffer, A., Mange, D., Rossier, J.: Design of self-organizing bio-inspired systems. In: Arslan, T., et al. (eds.) Proceedings of the 2007 NASA/ESA Conference on Evolvable Adaptative Hardware and Systems (AHS 2007), pp. 413–419. IEEE Computer Society, Los Alamitos (2007)
4. Stauffer, A., Mange, D., Rossier, J.: Self-organizing Systems Based on Bio-inspired Properties. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 1171–1181. Springer, Heidelberg (2007)
5. Upegui, A., Thomas, Y., Sanchez, E., Perez-Uribe, A., Moreno, J.-M., Madredas, J.: The Perplexus bio-inspired reconfigurable circuit. In: Lohn, J., et al. (eds.) Proceedings of the NASA/ESA Conference on Adaptative Hardware and Systems (AHS 2007), pp. 600–605. IEEE Computer Society, Los Alamitos (2007)

Epigenetic Tracking: Biological Implications

Alessandro Fontana

IEEE
alessandro.fontana@ieee.org

Abstract. "Epigenetic Tracking" is an evo-devo method to generate arbitrary 2d or 3d shapes; as such, it belongs to the field of "artificial embryology". The objective of this paper is to explore the implications of the method for some relevant aspects of biology, namely junk DNA, the "ontogeny recapitulates philogeny" theory and the process of ageing. After presenting the latest experiments performed with 3d target shapes, the mentioned aspects are investigated and the explanation provided by Epigenetic Tracking is discussed.

1 Artificial Embryology and Epigenetic Tracking

The previous work in the field of artificial embryology falls into two broad categories: the grammatical approach, originated by Lindenmayer (Lindenmayer, 1968) and the cell chemistry approach, that draws inspiration from the early work of Turing (Turing, 1952). Notable examples of grammatical embryogenies are (Hornby and Pollack, 2002), (Cangelosi et al., 2003) and (Gruau et al., 1996). Among cell chemistry embryogenies, we recall (Kauffman, 1969) and, more recently, (Bongard and Pfeifer, 2001), (Miller and Banzhaf, 2003) and (Doursat, 2007). "Epigenetic Tracking" (E.T.) is the name of an embryogeny applied to morphogenesis, i.e. the task of generating arbitrary 2d or 3d shapes (Fontana, 2008). In the present work, after reporting the results of the last experiments performed with 3d shapes, we will explore the implications of E.T. for some relevant aspects of biology: junk DNA, the "ontogeny recapitulates philogeny" theory and the process of ageing.

2 Experiments

In this section we report the results of the latest experiments, conducted with some 3d black-and-white target shapes (figure 1); all targets have a total number of cells ranging from 50.000 to 100.000, have been evolved with a Genome composed of 200-300 instructions, in a number of generations around 20.000; to our knowledge, no other method is able, by means of evo-devo techniques, to generate target shapes with this size and variety. The time required to evolve a shape grows linearly with the number of cells: as a result, any shape can be considered within the reach of the method, that can therefore be proposed as a possible solution to the problem of evo-devo morphogenesis. The remainder of

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 10–17, 2011.

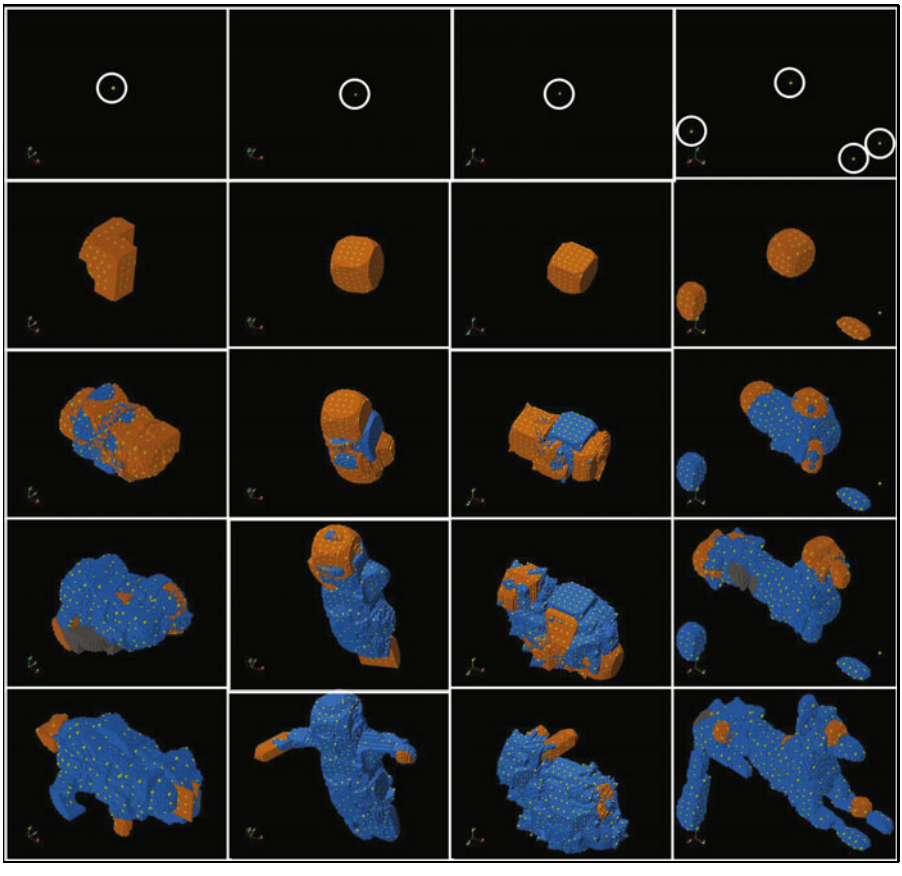


Fig. 1. some development steps of the triceratops, the child, the Rufa bunny and anubis; all shapes developed from single zygotes (with the exception of anubis, developed from four zygotes)

this paper will be dedicated to investigating the method's implications for key topics such as junk DNA, the "ontogeny recapitulates philogeny" theory and the process of ageing; junk DNA will be examined first.

3 Biological Implications

Junk DNA. In molecular biology, "junk DNA" is a collective label for the portions of the DNA sequence of a genome for which no function has been identified. About 95% of the human genome has been designated as "junk", including most sequences within introns and most intergenic DNA. Some chromosomal regions are composed of the now-defunct remains of ancient genes known as *pseudogenes* (which were once functional copies of genes but have since lost their

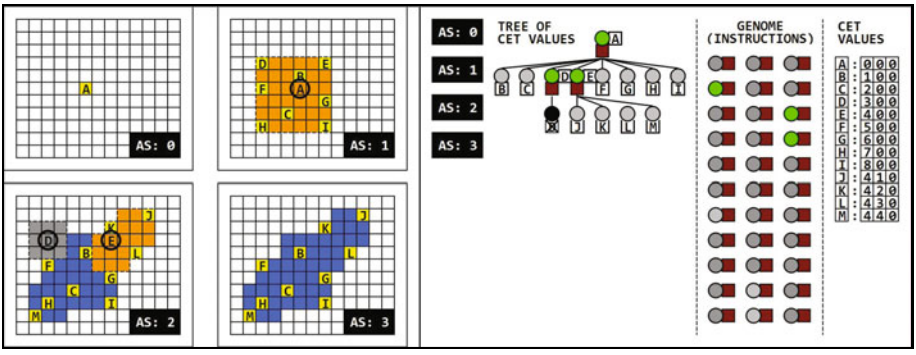


Fig. 2. Left: example of development in four steps steered by three instructions. Right: the corresponding tree of CET values. Grey circles represent CET values that do not match any instructions and instructions' left parts that do not match any CET values (junk elements); green circles represent CET values and instructions' left parts that match; brown squares represent instructions'right parts.

protein-coding ability) and as much as 25% of the human genome is recognisably formed of *retrotransposons*. There are some hypotheses for how junk DNA arose and why it persists in the genome, but none is conclusively established.

In E.T., at any given moment through the course of evolution, the set of all CET values generated during an individual's development (called "tree of CET values", abbreviated TCV) can be divided into i) CET values that activate an instruction during development and ii) CET values that *do not* activate any instruction during development. In the same way the individual's Genome is composed by i) instructions that become active during development and ii) instructions that *do not* become active during development. By analogy with real genomes, elements in the two categories labelled with ii) can be defined as junk CET values and junk instructions respectively. A schematic representation of this distinction is given in figure 2, that on the left shows an example of development in four steps and on the right shows the corresponding TCV and Genome (grey circles represent junk elements). For ease of reference the set of CET values active during development will be called TCV-D (D for development), while the set of CET values not active during development will be called TCV-I (I for inactive); analogously, the set of instructions active during development will be designated with GEN-D and the set of instructions not active during development will be designated with GEN-I. We will argue that the presence of junk in both the tree of CET values and the Genome is an inescapable phenomenon, intimately linked to the functioning of the epigenetic tracking machine.

The variable CET, as we know, is structured as an array of N integers, where N is the number of age steps. An alternative choice would be to use a scalar CET value, along with a global counter, in the following way. Each time a proliferation event takes place anywhere in the shape, the CET value assigned to the first new driver cell is the value held by the global counter; subsequent

values are determined adding one at each assignment and the global counter is updated correspondingly, such that it always holds the first value not assigned (being zero the zygote's CET value). The major drawback of this apparently simpler approach is the need for a global variable (the global counter), that has to be accessible anytime by any cell to be updated, which makes it not biologically plausible. The introduction of an array version for the CET eliminates the need of a global variable, restoring decentralisalised organisation and biological plausibility.

Unfortunately, this solution brings about another problem: passing from a single integer to an array of integers increases the size of the GA search space, such that, for values of N not too small, evolution comes to a halt. The countermeasure consists in a procedure called "Germline Penetration": such procedure acts on the Genome of each individual at the end of development, copying at random (some) CET values occurred during development onto left parts of instructions in the Genome, as a "suggestion" for the GA for where to search in the subsequent generation: with this procedure in place, the effectiveness of the GA is restored. To avoid disrupting development, the copied instructions are set as inactive, so that they start their "career" as junk instructions.

With the previous considerations in mind we can now turn our attention to junk DNA. The epigenetic tracking machine, the way it is conceived, cannot do its job without generating a lot of junk CET values: in our experiments, the average ratio (CET values used / CET values generated) is around 5% (in the development of anubis, for instance, 1890 CET values are generated and only 74 are used). The presence of such a high percentage of unused CET values looks like a waste of resources and we could ask whether there is a way to reduce it. The most straightforward way to reduce the amount of unused CET values consists in decreasing the ratio between driver and normal cells in proliferation events: for instance, instead of the value of 1:125 used in our experiments, we could use a value of 1:500 or lower. This would cause the shape to have, at any step, a sparser distribution of driver cells (a lower density of "yellow dots") and, as a result, the "sculpting" would become less precise and the evolution of development would become harder.

So, there seems to be a trade-off between the precision of sculpting and the density of driver cells (and hence the percentage of junk CET values): the second aspect cannot be improved without worsening the first. Since the effectiveness in evolving shapes is the primary objective of the method, in this regard we are not prepared to make concessions: therefore, we must accept a certain amount of unused CET values. The result of the discussion so far can be expressed by saying that the following two facts have emerged as inherent, inescapable characteristics of our method: i) the presence of a high percentage of junk CET values, to allow for a sufficient precision of the sculpting process and ii) the need to use the procedure called Germline Penetration, to allow the GA to work with an array-structured CET variable (thus eliminating the drawbacks deriving from the use of a scalar CET variable). Putting these two elements together, it would not be surprising to observe (as we did in our experiments) that Germline

Penetration acts like a shuttle, transferring junk CET values from the TCV onto a corresponding number of junk instructions' left parts in the Genome (with the hope that they meet each other and will not be junk anymore!). In conclusion, the presence of junk material in both the TCV and the Genome (the second mirroring the first) is inescapably connected to the core of the epigenetic tracking machine, a requirement essential to its *evolvability*.

We conclude this subsection with two observations of a more speculative nature. Firstly, should we hypothesise the existence of a mechanism akin to Germline Penetration also in biological systems, we would be naturally led to think of mobile DNA elements, or transposons, as the actual device used to carry the CET values from the biological equivalent of driver cells, spread throughout the organism, to the germline cells, where they would deliver the recipe of the current development as a suggestion for future improvements. Secondly, we note that, by allowing the developmental history of an organism to influence its genome and therefore to be passed on to the subsequent generation, the mechanism of Germline Penetration adds a Lamarckian touch to the Darwinian evolution implemented by the Genetic Algorithm.

"Ontogeny Recapitulates Phylogeny". In 1866, the German zoologist Ernst Haeckel proposed that the embryonal development of an individual organism (its ontogeny) followed the same path as the evolutionary history of its species (its phylogeny); in other words, Haeckel's **recapitulation theory** (also known as "ontogeny recapitulates phylogeny") claims that the development of advanced species passes through stages represented by adult organisms of more primitive species. Although modern biology rejects the literal and universal form of Haeckel's theory, the basic idea of recapitulation is still widespread.

In the epigenetic tracking framework, we have just seen that, at the end of the development of a species X (consider for instance the last step -the Nth- in the development of the triceratops in figure 1), many driver cells are present (the yellow dots in the figure) that have not been activated during development: we called their CET values junk CET values. If, in a subsequent generation, some of these driver cells get activated by effect of genomic mutations (i.e. new instructions appear in the Genome that match their CET values), a corresponding number of changes will affect the shape in the subsequent age step (the N+1th, not present for species X), giving rise to a new species, Y. The consequence of this evolutionary jump is that the development of species Y will pass through a step (the Nth) that happens to coincide with the last step of the development of species X (still the Nth), a sentence that could indeed be taken as a formal description of the "ontogeny recapitulates phylogeny" theory.

Ageing. Ageing is the accumulation of changes in an organism over time, leading to a steady decline in bodily functions: we will now show how E.T. can account for the ageing phenomenon. As we know, for a given individual development unfolds in N age steps; at the end of it the individual's fitness is evaluated and right afterwards the Genome content is handed over, after being processed by the genetic operators, to the subsequent generation. The moment of fitness

evaluation, that in nature corresponds roughly to the moment of reproduction, has always coincided in our experiments with the end of the simulation; on the other hand, we can imagine to let the global clock AS tick on and see what happens, in the period after the moment of fitness evaluation. The distinction between the periods (i.e. the sets of age steps) before and after fitness evaluation can be thought to correspond to the biological periods of development (say, until 25 years of age in humans -the average age of reproduction) and ageing (from 25 years of age onwards); by analogy, such periods will be called "artificial development" and "artificial ageing" respectively.

As pointed out several times, at the end of an individual's development many junk CET values are present, as well as many junk instructions; such stock of junk represents a reservoir of events that can potentially be triggered after the moment of fitness evaluation, in the period that we called artificial ageing. Since these events occur after fitness evaluation, they are by definition not affecting the fitness value; for this reason they will tend to have a random nature and their effects on the overall fitness of the phenotype will be more likely to be detrimental than beneficial: they can be thought of as a random noise superimposed on the phenotype created by the instructions subject to evolutionary pressure (in nature, actually, an individual's fitness does not depend only on characteristics manifesting themselves before reproduction, but also on characteristics appearing after reproduction, as also those can affect the survival chances of its progeny; in other words the effect of changes on the fitness tends to decrease as the age of their appearance increases, rather than going abruptly to zero right after reproduction: we chose to ignore this "subtlety" for simplicity reasons).

The set of CET values not active during development (TCV-I) can therefore be further subdivided into a set of CET values that become active during the ageing period (TCV-A, A for ageing) and a set of CET values that are never active (TCV-J, J for junk), so that TCV = TCV-D U TCV-A U TCV-J (analogous distinction can be done also for the Genome). An example of artificial ageing is reported in figure 3 for a "face" shape (picture of 100x100 size with 16 grey shades); the left part shows steps 0-9, belonging to the period of development: the shape grows from the single cell stage to the mature phenotype in step 9, when fitness is evaluated; the right sequence refers to the period of ageing (steps 10-19), characterised by the accumulation of random events, whose global effect causes a progressive deterioration of the quality of the image. In conclusion, the phenomenon of artificial ageing can be thought of as the result of the cumulative action of instructions activated after the moment of fitness evaluation, on which natural selection has no (less) effect.

We wish to conclude this section dedicating a final comment to the role played by junk in the phenomenon of artificial ageing. In previous paragraphs the sets of CET values/instructions inactive during development, indicated with TCV-I and GEN-I respectively, were shown to be a useful reservoir of instructions and an indispensable tool to explore new evolutionary paths. In this subsection we have shown how a part of it (TCV-A and GEN-A) is actually devoted to cause random events that manifest themselves after the moment of fitness evaluation,

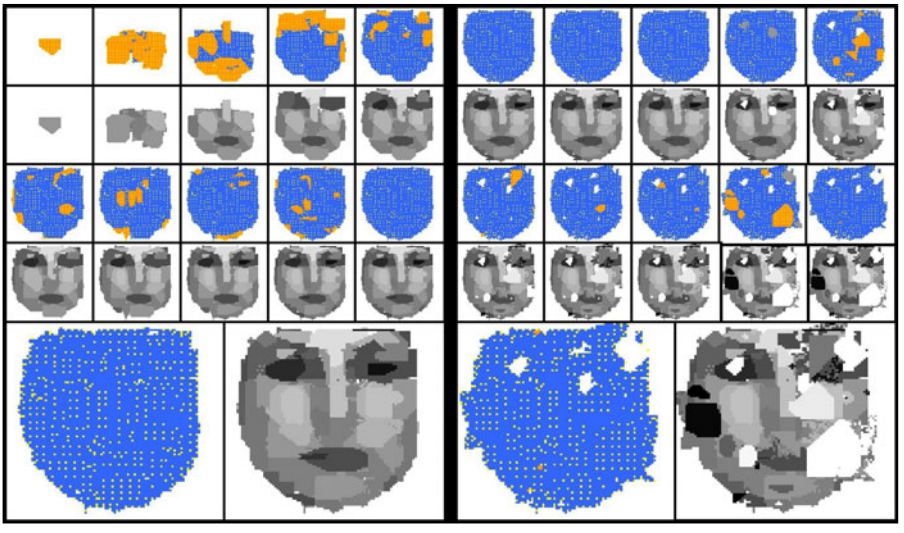


Fig. 3. The "face". On the left the period of development (steps 0-9): the shape grows from a single cell to the mature phenotype in step 9, fitness is evaluated; on the right the period of ageing (steps 10-19): the picture quality deteriores steadily under the action of random instructions.

relegating to TCV-J and GEN-J the role of true junk. On the other hand, we notice how the border between TCV-A and TCV-J (GEN-A and GEN-J) is permeable (elements can move between the two sets) and the average size of TCV-A (GEN-A) is proportional to the size of TCV-J (GEN-J). These considerations bring us to deducting, in the epigenetic tracking "world", a direct link between the evolvability of a species and its susceptibility to ageing, being both aspects mediated by the presence of a big stock of junk. The fact that bats have unusually small genomes (i.e. little junk) and display a remarkably long lifespan (i.e. they appear to age less) among mammals of comparable dimension (Van den Bussche et al., 1995), could hint to the existence of a similar link also in real biological systems.

4 Conclusions

In this work we have presented the last experiments performed with a method called Epigenetic Tracking on 3d shapes. The method has subsequently been utilised to interpret some key biological phenomena, namely junk DNA, the "ontogeny recapitulates phylogeny" theory and ageing, that have sparked much speculation over the last decades. Epigenetic Tracking has been shown able to provide simple and straightforward explanations for such phenomena, being thus validated as a useful theoretical biology tool in the evo-devo context. Finally, I take the opportunity to thank my friend Perry for reviewing this paper.

References

Bongard, G., Pfeifer, R.: Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). Morgan Kaufmann, San Francisco (2001)

Cangelosi, A., Nolfi, S., Parisi, D.: Artificial life models of neural development. In: Press, A. (ed.) On Growth, Form and Computers (2003)

Doursat, R.: The self-made puzzle: Integrating self-assembly and pattern formation under non-random genetic regulation. In: 7th International Conference on Complex Systems, ICCS (2007)

Fontana, A.: Epigenetic tracking, a method to generate arbitrary shapes by using evo-devo techniques. In: Proceedings of the 8th International Conference on Epigenetic Robotics, EPIROB (2008)

Gruau, F., Whitley, D., Pyeatt, L.: A comparison between cellular encoding and direct encoding for genetic neural networks. In: Proceedings of the First Annual Conference on Genetic Programming (1996)

Hornby, G.S., Pollack, J.B.: Creating high-level components with a generative representation for body-brain evolution. Artificial Life 8(3)(2002)

Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22 (1969)

Lindenmayer, A.: Mathematical models for cellular interaction in development. Journal of Theoretical Biology 18 (1968)

Miller, J.F., Banzhaf, W.: Evolving the program for a cell: from french flags to boolean circuits. In: Press, A. (ed.) On Growth, Form and Computers (2003)

Turing, A.: The chemical basis of morphogenesis. Philosophical Transactions of the Royal Society 237 (1952)

Van der Bussche, R.A., Longmire, J.L., Baker, R.J.: How bats achieve a small c-value: frequency of repetitive dna in macrotus. Mammalian Genome 6(8) (1995)

The Effect of Proprioceptive Feedback on the Distribution of Sensory Information in a Model of an Undulatory Organism

Ben Jones[1], Yaochu Jin[2], Bernhard Sendhoff[2], and Xin Yao[1]

[1] School of Computer Science, University of Birmingham, UK
[2] Honda Research Institute Europe GmbH, Germany

Abstract. In an animal, a crucial factor concerning the arrival of information at the sensors and subsequent transmission to the effectors, is how it is distributed. At the same time, higher animals also employ proprioceptive feedback so that their respective neural circuits have information regarding the state of the animal body. In order to disseminate what this practically means for the distribution of sensory information, we have modeled a segmented swimming organism (animat) coevolving its nervous system and body plan morphology. In a simulated aquatic environment, we find that animats artificially endowed with proprioceptive feedback are able to evolve completely decoupled central pattern generators (CPGs) meaning that they emerge without any connections made to neural circuits in adjacent body segments. Without such feedback however, we also find that the distribution of sensory information from the head of the animat becomes far more important, with adjacent CPG circuits becoming interconnected. Crucially, this demonstrates that where proprioceptive mechanisms are lacking, more effective delivery of sensory input is essential.

Keywords: animat, morphology, neural control, proprioception, behaviour.

1 Introduction

The state of a given animal's external environment or niche, is presented to the animal via its sensory system. This generates informational cues regarding for example, predator or prey items, allowing the animal's nervous system to invoke either pervasive or evasive behaviours. Typically over time, the animal is able to learn and adapt[1]. Higher animals also employ proprioceptive mechanisms enabling them to detect the current state of the locomoting body, serving as a sensory feedback mechanism for the underlying neural circuits. Previous studies have shown that central pattern generators (CPGs) responsible for the periodic movement control are all affected and constrained by such feedback, e.g. [13]. Other studies have highlighted how feedback can help undulatory organisms surpass a 'speed barrier' [7,8]. The necessity of proprioception in the peristaltic movements of drosophila larvae has also been established, without which, locomotion is seen to be significantly degraded [17]. Typically such proprioceptive mechanisms are 'stretch receptors' within the animal's body wall, e.g. [6]. In

[1] Note that in this paper, we have no concept of learning, rather behaviour is considered only in reactive 'braitenberg vehicle' terms [4].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 18–26, 2011.

order for the animal to respond correctly, all of this sensory information has to reach the appropriate effectors.

We pick up on the point of proprioceptive feedback and its influence on sensory information distribution. We model a segmented three dimensional aquatic organism with movement mechanisms not dissimilar to the vertebrate lamprey. In an initial experiment, the animat is endowed with a proprioceptive mechanism whilst in the second, it is not. In both, the animat has an abstract visual system which it may or may not utilise depending on how the neural circuits become interconnected. The goal is for the animat to swim forwards towards a predefined target.

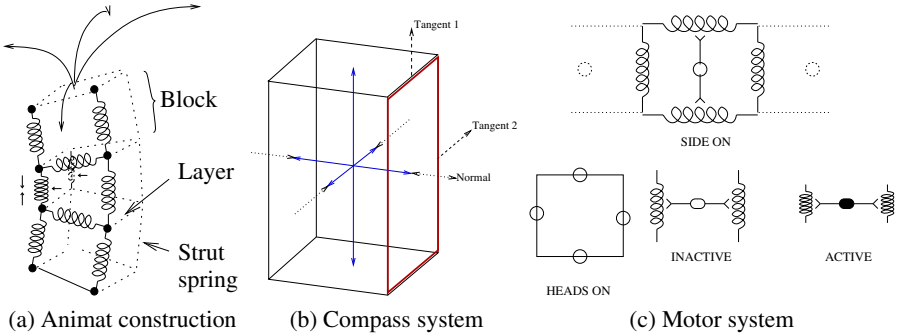
Whilst the field of physically realistic locomotion is old (see [9] for a review), the incorporation of some abstract visual system is novel. Beauregard and Kennedy model a 2D lamprey able to undertake tracking of a moving object [2]. Indeed, the visual system that their model utilises provides a basis in our model. Ijspeert models a visual system in a 3D simulated salamander able to track a moving object both in land and water [10]. In Biology, Deliagina et al. have found activity differences in the reticulospinal neurons, a system within the lamprey transmitting signals from the brain to the spinal cord, whenever the lamprey turns [5]. This highlights the functional significance of effective information distribution from sensors to effectors. The rest of this paper is laid out as follows. Section 2 gives an overview of the simulation environment. Section 2.3 provides experimental details. Section 3 presents our main findings. We conclude in Section 4.

2 Simulation Environment

The simulation environment has been implemented in C++. There are 2 main components making up the system: the animat and the evolutionary setup. They are explained below and an overview of the experimental setup also follows.

2.1 Animat

Geometry. The animat is soft-bodied being entirely constructed out of springs. These springs are connected together to form cuboids which are then themselves connected together to form the overall morphology, Fig. 1a.



(a) Animat construction (b) Compass system (c) Motor system

Fig. 1. Animat geometry. The compass system is used to derive water forces. The motor system shows how motor neurons 'contract' springs in pairs.

The water force model. An external water force is applied to each face of a given animat block. The force is derived from the velocity of the face which is taken to be the average velocity of all four point masses, similar to, e.g. [16]. The force is computed by initially splitting the velocity vector into its three components (as highlighted in Fig. 1b):

$$t_1 = \hat{\mathbf{t_1}} \cdot \mathbf{v} \qquad t_2 = \hat{\mathbf{t_2}} \cdot \mathbf{v} \qquad n = \hat{\mathbf{n}} \cdot \mathbf{v} \tag{1}$$

where $\hat{\mathbf{t_1}}$, $\hat{\mathbf{t_2}}$ and $\hat{\mathbf{n}}$ are normalised tangent and vector components of the block face and $\mathbf{v}$ is the velocity of the face. We then compute the three force components as follows:

$$\Xi(t_1) = -\gamma_{t_1} \text{sgn}(t_1)(t_1)^2 \tag{2}$$
$$\Xi(t_2) = -\gamma_{t_2} \text{sgn}(t_2)(t_2)^2 \tag{3}$$
$$\Xi(n) = -\gamma_n \text{sgn}(n) n^2 \tag{4}$$

where the γ parameters control the levels of application of each of the three components. The actual water force, $\mathbf{w}$, that can be applied to each of the four point masses making up the block face is calculated as follows:

$$\mathbf{f} = \Xi(t_1)\hat{\mathbf{t_1}} + \Xi(t_2)\hat{\mathbf{t_2}} + \Xi(n)\hat{\mathbf{n}} \tag{5}$$

$$\mathbf{w} = \mathbf{f} c d A \tag{6}$$

where c is a viscosity coefficient, d is drag and A is the area of the block face. Note that in our model we have set c and d to 1 since it is sufficient to tune the γ parameters.

Neural system. The neural system is based on a continuous time recurrent neural network. The membrane potential, u_j, of a neuron is modelled as follows [3]:

$$\frac{du_j}{dt} = \frac{1}{\tau_j}\left(-u_j + \sum_{i=1}^{C} w_{ji} a_i + I_j\right) \tag{7}$$

where τ_j is a time constant, w is a vector of presynaptic connection weights and I_j is an external input current. The value a_i is a presynaptic neuron's membrane activity computed as follows:

$$a_i = \begin{cases} \tanh(u_i - \beta_i) & |u_i| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Note that given Eq. 8, the function is only employed if the neuron's membrane potential is not 0. We have this restriction in order to ensure that neurons need some initial input, for example, from a sensor, before they can generate any kind of dynamic. Without it, a neuron would always potentially have an activity, because the bias value, β_j, would allow for this. The weight values are computed from the interneuronal Euclidean distance as in [11,12]. Connectivity also comes about as a function of distance according to the sigmoid,

$$\sigma(\lambda, s, d_{ij}) = \frac{2}{2 + \exp((\lambda/s) * d_{ij})} \tag{9}$$

where λ is an evolved parameter, s is a scaling parameter set to 4.5 and d_{ij} is the Euclidean distance between neurons *i* and *j*. A connection is established if the function produces a value >0.5.

Motor system. Each motor is an excitatory neuron. Being position-fixed, it is also considered part of the body plan. Each animat block has 4 motors, 1 associated with each face of the block. A given motor actuates a vertical spring-pair of the block face, see Fig. 1c. The amount of force applied to a spring pair is proportional to the membrane potential of the associated motor neuron.

Sensory system. The animat has a very rudimentary sensory system consisting of 4 sensory neurons that remain position-fixed at the head of the animat (one at the top-middle of each block face). Current is injected only into the closest sensor from the target and is inversely proportional to the angle of the target from the given sensor. Whilst there are no turning constraints required in our later experiments, this setup paves the way for future experimentation. The input current injected into the closest sensory cell is thus: $I_s = \exp(\phi + 0.01)$.The value 0.01 ensures that there will be some input current, even when the target angle, ϕ, is 0. Note that this sensory mechanism is partially based on the exponentiated bearing-based tracking model employed in [2].

Proprioceptive feedback mechanism. The proprioceptive mechanism is based on a notion of stretch receptor activity, for example, that found in the leech [6]. Also, as with the sensory system outlined above, the proprioceptive mechanism is exponentiated taking the amount of side spring distension as input (difference in length of spring from resting length). This input current, I_M, is then fed directly into the associated motor neuron computed as $I_M = \exp(\Delta d)$ where Δd is the level of spring distension.

2.2 Evolved Components

A mixed real-valued and Boolean evolutionary algorithm having discrete recombination, self-adaptive mutation (see [1]) and tournament selection with an elitist strategy is used to evolve a genotype consisting of three main components: the body-plan, the neural architecture, and the neural properties.

Body-plan. In the simulation, we consider the number of body segments, the length of each segment and the symmetry of the active motor configuration (refer to [11] for details of this latter aspect) to all be parts of the body-plan morphology. Note also that when the length of a segment changes, the neural distribution's spread within that segment commensurately changes.

Neural architecture. Inside of each body plan segment, there are 6 interneurons, the polar coordinate positions of which are randomly initialised and subsequently evolved. Secondly, a set of λ values tuning the connectivity function as given in Eq. 9 are also evolved depending on the type of connectivity: λ_{II}, λ_{IE}, λ_{SE}, λ_{AA} where I=interneuron, E=effector neuron, S=sensory neuron; AA indicates connections between interneurons in adjacent segments.

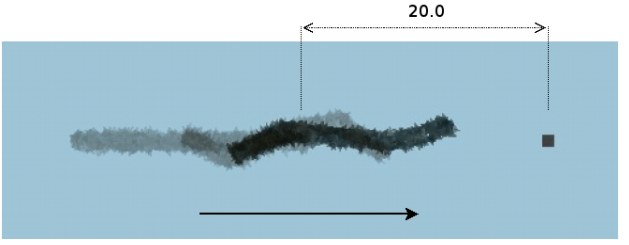


Fig. 2. A sequence of overlaid screenshots at behavioural iterations 1, 245 and 384 (a total of 400 are permitted) for an evolved animat. The animat's task is to swim towards the cube in the direction indicated by the lower arrow.

Neural properties. These include the neuron time-constants, thresholds and whether or not a neuron is inhibitory. A weight value between a neuron pair is derived according to the distance between them, as in [11, 12].

2.3 Experimental Overview

Our experiments address how sensory information should be distributed when we consider proprioceptive mechanisms, especially in view of connectivity patterns that might emerge between different neural circuits. We have therefore conducted two sets of 30 experiments for statistical significance with each individual experiment being allowed to run for 500 generations. In the first setup, the animat is endowed with proprioceptive feedback. In the second, it is not. In both, the animat is required to swim forwards in order to reach a pre-defined target. The fitness function is simply $f_1 = 20.0 - d_{target,animat}$, see Fig. 2.

3 Results

In Fig. 3, we can see that animats endowed with the proprioceptive mechanism performed significantly better than those that were not. We can secondly observe, that a

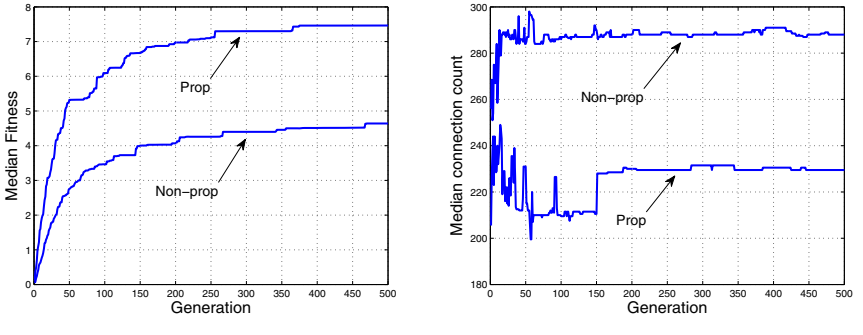
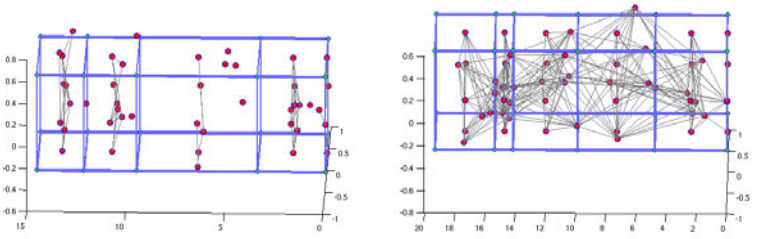


Fig. 3. Comparison of fitness medians and connectivity count medians for proprioceptive and non-proprioceptive variants



(a) Proprioceptive (b) Non-proprioceptive

Fig. 4. Representative architectures to have evolved for each setup. For the proprioceptive individual, the architecture consists of decoupled neural circuits, with sparse connectivity. By contrast, connectivity is far higher in the non-proprioceptive individual.

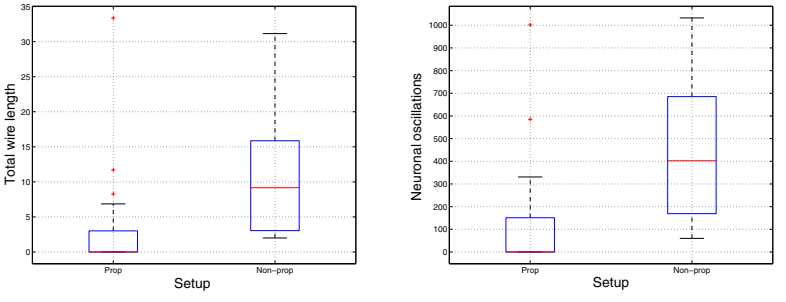


Fig. 5. Box plots of wire lengths and neural oscillations to have emerged for the best individuals. Both properties are observed to be of higher magnitude for the non-proprioceptive individuals.

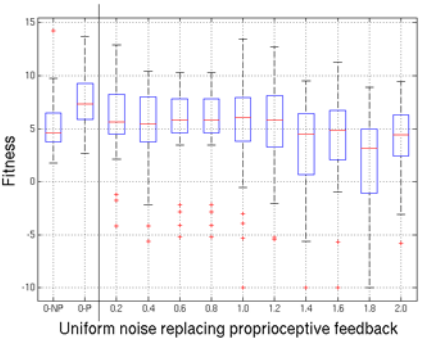


Fig. 6. A boxplot showing how differing amounts of uniform noise used to replace the proprioceptive feedback mechanism, affects fitness. The '0-NP' and '0-P' cases left of the vertical line are representative of fitness values for the non-proprioceptive and proprioceptive individuals without such noise. Those to the right are representative of fitness values for the proprioceptive individuals, after noise has replaced the normal feedback mechanism.

higher number of connections were required in those animats without the proprioceptive mechanism. A higher number of connections equates to higher connectivity between neurons in *adjacent* neural circuits, higher connectivity within *individual* neural circuits, and, the presence of some connections from the *sensory neurons* located in the head of the animat. Representative neural architectures depicting two such animats are given in Fig. 4. Finally, we relate such architectural distribution to total wire length and total number of neuronal oscillations observing that for the animats without proprioceptive feedback, there is a marked increase in total wire length, see the left panel of Fig. 5. This is to be expected given our prior observations regarding increased connectivity. We secondly find that animats without proprioceptive mechanisms generate a higher number of neuronal oscillations, refer to the right panel of Fig. 5.

Although it would appear that proprioception as it exists in our model benefits the animat behaviour, it is difficult to know with any certainty whether the feedback mechanism is truly serving to modulate the neuronal dynamics as would be the case in true proprioception, or, whether it is just triggering the network to reach a particular attractor state. In order to test this, we have performed a final set of experiments taking the 30 best proprioceptive individuals and replacing the feedback mechanism with varying levels of noise. As mentioned in the model description (subsection 2.1), this feedback mechanism works by injecting into an associated motor neuron, an input current that is proportional to the level of spring distension. Replacing it with a uniform noise simply substitutes the input current for a float value generated from the range [-n,n]. The smallest level of noise chosen was [-0.2,0.2], whilst the largest was [-2,2]. If the feedback is only serving to trigger the network then we can expect the animat to be robust to arbitrary value. The results are presented in Fig. 6. We can see that up to a noise range of [-1.2,1.2] the performance is slightly degraded but all performances up to this point are approximately equal, whilst a noise range greater or equal to [-1.4,1.4] sees a degradation in animat performance.

4 Discussion and Conclusions

We have observed that proprioception advances the animat's ability to locomote forwards. Also, that replacing the proprioceptive mechanism with noise up to a point does little in degrading undulatory behaviour (from [-0.2,0.2] to [-1.2,1.2]). This suggests how under normal non-noisy circumstances, the feedback might be serving as a 'triggering' mechanism. However if the noise is advanced to too great a level (from [-1.4,1.4] to [-2.0,2.0]), performance is seen to be degraded. This firstly suggests that the system is not robust to high levels of this type of noise; secondly, that correct proprioception has a fitness enhancing or at least a modulating effect on the neural dynamics given that any level of noise is seen to degrade performance.

Interestingly, with proprioception, the neural architecture often evolves such that the individual neural circuits become completely decoupled. Inferably, this stems from a need to reduce interference between the neural circuits so that correct oscillatory dynamics can result. Indeed, by artificially adding interconnections, performance is degraded (results not shown). Therefore in some cases, centralized control, or interconnectivity between individual CPG 'modules', can be detrimental.

Thirdly, since in the non-proprioceptive variant there are no feedback mechanisms, the neural system has no way of directly relying on body-shape information. Yet in the proprioceptive case, the actual body is allowed to become part of the process that yields behaviour. This is fundamental because the neural system then has a direct informational link to the body, so strengthens the *passive role of the body-physics* ('morphological computation', [14]). This is evident on two levels. Firstly, the number of neural oscillations which is lower in the proprioceptive case allows us to speculate that more of the CPG dynamic could be offloaded to the passive body movements, see Fig. 5. Secondly, sparser neural connectivities in the proprioceptive case inferably demonstrates less of a need for complex neural processing; when feedback mechanisms are available, computation can be more so aided by passive body physics thus allowing for a reduction in neural computation. Note also, a higher level of connectivity in the non-proprioceptive case may conversely compensate for a lack of sensory feedback.

There are a number of major extensions that we envisage. The first involves analysis of the different body plan components, for example body plan segment length and number of body-plan segments, to see how they affect the performance of the animat. Work has begun on this. A second undertaking will then incorporate an energy measure as we did for a model of a radially symmetric organism in [12]. Finally, we might incorporate a developmental process so that the model more realistically reflects biological systems. The work of Schramm et al., is an interesting start in this direction [15].

References

1. Bäck, T., Schwefel, H.-P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation 1(1), 1–23 (1993)
2. Beauregard, M., Kennedy, P.J.: Robust simulation of lamprey tracking. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 641–650. Springer, Heidelberg (2006)
3. Blynel, J., Floreano, D.: Levels of dynamics and adaptive behavior in evolutionary neural controllers. In: From Animals to Animats 7: Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior. MIT Press, Cambridge (2002)
4. Braitenberg, V.: Vehicles, Experiments in Synthetic Psychology, Cambridge, Mass. (1984)
5. Deliagina, T.G., Zelenin, V., Fagerstedt, P., Grillner, S., Orlovsky, G.N.: Activity of reticulospinal neurons during locomotion in the freely behaving lamprey. Journal of Neurophysiology 83, 853–863 (2000)
6. Friesen, W.O., Kristan, W.B.: Leech locomotion: swimming, crawling, and decisions. Neurobiology of Behaviour 17, 704–711 (2008)
7. Grillner, S., Kozlov, A., Dario, P., Stefanini, C.: Modeling a vertebrate motor system: pattern generation, steering and control of body orientation. Progress in Brain Research 165 (2007)
8. Ijspeert, A.J., Hollam, J., Willshaw, D.: Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. Adaptive Behavior 7(2), 151–172 (1999)
9. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: a review. Neural Networks 21(4), 642–653 (2008)
10. Ijspeert, A.J., Arbib, M.: Visual tracking in simulated salamander locomotion. In: Sixth International Conference of The Society for Adaptive Behavior (SAB 2000), Paris, pp. 88–97 (2000)

11. Jones, B., Jin, Y., Sendhoff, B., Yao, X.: Evolving functional symmetry in a three dimensional model of an elongated organism. In: Proceedings, ALife XI, Winchester, UK, pp. 305–312 (2008)
12. Jones, B., Jin, Y., Yao, X., Sendhoff, B.: Evolution of neural organization in a hydra-like animat. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008. LNCS, vol. 5506, pp. 216–223. Springer, Heidelberg (2009)
13. Nishii, J.: A learning model of a periodic locomotor pattern by the central pattern generator. Adaptive Behavior 7(2), 137–149 (1999)
14. Pfeifer, R.: Morphological computation: Connecting brain, body, and environment. In: Ijspeert, A.J., Masuzawa, T., Kusumoto, S. (eds.) BioADIT 2006. LNCS, vol. 3853, pp. 2–3. Springer, Heidelberg (2006)
15. Schramm, L., Jin, Y., Sendhoff, B.: Emerged coupling of motor control and morphological development in evolution of multi-cellular animats. In: Kampis, G., Karsai, I., Szathmáry, E. (eds.) ECAL 2009, Part I. LNCS, vol. 5777, pp. 25–32. Springer, Budapest (2009)
16. Sfakiotakis, M., Tsakiris, D.P.: Simuun: A simulation environment for undulatory locomotion. International Journal of Modelling and Simulation (2006)
17. Song, W., Onishi, M., Jan, L.Y., Jan, Y.N.: Peripheral multidendritic sensory neurons are necessary for rythmic locomotion behaviour in drosophila larvae. PNAS 104(12), 5199–5204 (2007)

Emerged Coupling of Motor Control and Morphological Development in Evolution of Multi-cellular Animats

Lisa Schramm[1], Yaochu Jin[2], and Bernhard Sendhoff[2]

[1] Technische Universität Darmstadt, Karolinenplatz 5, 64289 Darmstadt, Germany
[2] Honda Research Institute Europe, Carl-Legien-Str. 30, 63073 Offenbach, Germany

Abstract. A model for co-evolving behavior control and morphological development is presented in this paper. The development of the morphology starts with a single cell that is able to divide or die, which is controlled by a gene regulatory network. The cells are connected by springs and form the morphology of the grown individuals. The movements of animats are resulted from the shrinking and relaxation of the springs connecting the lateral cells on the body morphology. The gene regulatory network, together with the frequency and phase shifts of the spring movements are evolved to maximize the distance that the animats can swim in a given time interval. To facilitate the evolution of swimming animats, a term that awards an elongated morphology is also included in the fitness function. We show that animats with different body-plans emerge in the evolutionary runs and that the evolved movement control strategy is coupled with the body plan.

1 Introduction

Brain-body co-evolution has attracted much attention in the research field of artificial life [1] since the seminal work of Karl Sims [2]. The most attractive aspect of the work is that a developmental model using a directed graph has been adopted for both neural controller and body plan. However, no significant progresses to understand biological principles have been made since Sims' work due to the following two facts. First, the power of the models for brain-body co-evolution remains practically unchanged [3,4,5]. A biologically plausible model should be able to describe the biological development of both nervous system and body plan. Whereas models for either detailed modeling of neural development [6] or morphology [7] have been suggested, few models can achieve a balanced depth in modeling the development of both neural controller and body plan, and most of them are not able to perform biologically meaningful behaviors. Second, most work on brain-body co-evolution was meant mainly for improving the efficiency of generating a specific behavior, rather than understanding biological principles. An exception has been the work by Bongard and Paul [8], which studied the correlation between morphological symmetry and locomotive efficiency.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 27–34, 2011.

Most recently, increasing effects have been made to relate the research in brain-body co-evolution to biological principles. In [9], it is found that bilaterally symmetric body plan and neural architecture are favored in selection in a brain-body co-evolution of an elongated organism. The advantage of being able to evolve a bilaterally symmetric body plan or neural controller has been reported independently [10,11]. By taking energy efficiency into account in a hydra-like animat, it has been shown that a ring-like neural structure emerged in the animat [12], which is analogous to the nerve ring in biological hydra.

However, in the afore-mentioned models, the developmental process of the neural system and the body-plan is not included. To address this problem, we adapted a biologically plausible cell growth model [13] for neural development in a hydra-like animat with a fixed body plan [14]. In contrast to [6], the developed neural model is able to perform food grasping behavior by adjusting its connectivity and weights.

In this paper, we use a modified cell growth model for morphological development in a brain-body co-evolution environment, though behavior control is modeled in an abstract manner and evolved using a direct coding. Nevertheless, we believe that this work has made a solid step forward compared to [7] in that the developed morphology is able to perform a swimming behavior.

2 The Computational Model

The growth of the animat morphology is under the control of a gene regulatory network (GRN) and cellular physical interactions. The morphological development starts with a single cell put in the center of a two-dimensional computational area (80x80). Once the morphological development is complete, a controller is embedded in the morphology and the resulting swimming behavior is evaluated using a physics simulation engine.

2.1 Chromosome for Morphological Development

The cell growth model is slightly modified from the one in [13,14]. The model uses a GRN to regulate the developmental process. The GRN is defined by a set of genes consisting of regulatory units (RUs) and structural units (SUs). SUs define cellular behaviors, such as cell division and cell death, or the production of transcription factors (TFs) for intra- and inter-cellular interactions. The activations of the SUs are defined by the associated RUs, refer to Fig. 1. Note that single or multiple RUs may regulate the expression of a single or multiple SUs and that RUs can be activating (RU^+) or repressive (RU^-).

Each RU and TF has a certain affinity value that determines whether a TF can influence a RU. If the difference between the affinity values of a TF and a RU is smaller than a predefined threshold ϵ (in this work ϵ is set to 0.2), the TF can be bound to the RU to regulate. The affinity overlap ($\gamma_{i,j}$) between the i-th TF and j-th RU is defined by:

$$\gamma_{i,j} = \max\left(\epsilon - \left|\mathrm{aff}_i^{TF} - \mathrm{aff}_j^{RU}\right|, 0\right). \quad (1)$$

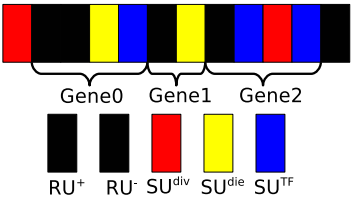


Fig. 1. An example chromosome for the development

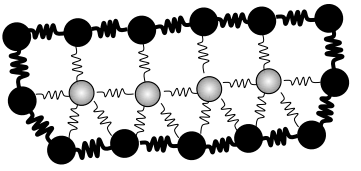


Fig. 2. Illustration of a body plan consisting of cells connected by springs. The springs at the outside of the body are able to change their natural length.

If $\gamma_{i,j}$ is greater than zero and the concentration c_i of the i-th TF is above a threshold (ϑ_j) defined in the j-th RU, then the i-th TF influences the j-th RU.

Thus, the activation level contributed by this RU (denoted by $a_j, j = 1, ..., N$) sums up to $a_j = \sum_{k=1}^{M} |c_k, -\vartheta_j|$, where M is the number of influencing TFs. The expression level of the k-th gene, that is regulated by N RUs, can be defined by

$$\alpha_k = 100 \sum_{j=1}^{N} h_j a_j (2s_j - 1), \tag{2}$$

where $s_j \in (0, 1)$. $2s_j - 1$ denotes the sign (positive for activating and negative for repressive) of the j-th RU and h_j is a parameter representing the strength of the j-th RU. If $\alpha_k > 0$, then the k-th gene is activated and its corresponding behaviors coded in the SUs are performed.

An SU that produces a TF ($\mathrm{SU}^{\mathrm{TF}}$) encodes all parameters related to the TF, such as the affinity value, the decay rate D_i^c, the diffusion rate D_i^f, as well as the amount of the TF to be produced:

$$A = \beta \frac{2}{1 + e^{-20 \cdot f \cdot \alpha}} - 1, \tag{3}$$

where f and β are both encoded in the $\mathrm{SU}^{\mathrm{TF}}$.

A TF produced by a SU can be partly internal and partly external. To determine how much of a produced TF is external, a percentage ($p^{\mathrm{ex}} \in (0, 1)$) is also encoded in the corresponding gene. Thus, $p^{\mathrm{ex}} \cdot A$ is the amount of external TF and $(1 - p^{\mathrm{ex}}) \cdot A$ is that of the internal TF.

External TFs are put on four grid points around the center of the cell, which undergo first a diffusion and then a decay process:

$$\text{Diffusion:} \quad \mathbf{u}_i(t) = \mathbf{u}_i(t-1) + 0.1 \cdot D_i^f \cdot (\mathbf{G} \cdot \mathbf{u}_i(t-1)), \tag{4}$$

$$\text{Decay:} \quad \mathbf{u}_i(t) = \min\left((1 - 0.1 \cdot D_i^c)\, \mathbf{u}_i(t),\, 1\right), \tag{5}$$

where $\mathbf{u}_i$ is a vector of the concentrations of the i-th TF at all grid points and the matrix $\mathbf{G}$ defines which grid points are adjoining.

In our experiments we put two prediffused, external TFs without decay and diffusion in the computation area. The first TF has a constant gradient in the x-direction and the second in the y-direction.

Table 1. Constants for the mechanical simulation environment

Mass of cells m	0.5	Short natural length of springs l_s	1.2
Radius of cells r	0.5	Minimal periodic time T_{min}	10
Damping constant d	1	Maximal periodic time T_{max}	400
Spring strength c	5	Simulation length t_{sim}	300.0
Normal natural length of springs l_n	2		

The SUs encode cellular behaviors and the related parameters. The SU for cell division encodes the angle of division, indicating where the daughter cell is placed. A cell with an activated SU for cell death dies at the end of the developmental timestep.

2.2 Chromosome for Motor Control

To embed a motor controller into the developed morphology, cells must be connected to a whole body plan. Cells are connected with a damped spring if the distance between them is smaller than 2.5. If a cell has fewer than two connections, it is then connected to its nearest neighbor to ensure morphological stability, see Fig. 2. The mechanical setups of the cells and springs are listed in Table 1.

The movement is defined by a change in the natural length of the springs connecting the lateral cells, as depicted in bold in Fig. 2. To ease the movement, the cell radius is set to 0.5 so that there is sufficient space between the cells for them to move. The natural length of the springs switches between l_n and l_s within one period T, which is subject to evolution. The morphology is split into 24 predefined segments and all springs in the same segment have the same phase shift ($\rho \in [0, T]$).

2.3 Physics Simulation Engine

We use BREVE [15] to simulate the behavior of the animats. In addition, we use a simple model for simulating the effects of water forces, which has also been adopted in [16]. In this model, the water forces for different elements i are computed as follows:

$$\boldsymbol{F}^i = \boldsymbol{F}^i_T + \boldsymbol{F}^i_N, \tag{6}$$

$$\boldsymbol{F}^i_T = -\lambda_T \cdot sgn(\boldsymbol{v}^i_T) \cdot (\boldsymbol{v}^i_T)^2, \tag{7}$$

$$\boldsymbol{F}^i_N = -\lambda_N \cdot sgn(\boldsymbol{v}^i_N) \cdot (\boldsymbol{v}^i_N)^2, \tag{8}$$

where λ_T and λ_N are the drag coefficients for each direction. λ depends on the effective area, a shape coefficient of the element and the fluid density. $\boldsymbol{v}^i_T$ and $\boldsymbol{v}^i_N$ are the velocities of element i in normal and tangential direction. We set $\lambda_T = 0.001$ and $\lambda_N = 2.5$ in this work. The water forces are computed for cells in the lateral of the body plan, represented by black circles in Fig. 2. The normal and tangential vectors of the body parts (i-th sphere) can be calculated by:

$$\boldsymbol{t}^i = \frac{\boldsymbol{p}^{i-1} - \boldsymbol{p}^{i+1}}{|\boldsymbol{p}^{i-1} - \boldsymbol{p}^{i+1}|} \tag{9}$$

$$\boldsymbol{n}^i = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \cdot \boldsymbol{t}^i, \tag{10}$$

where $\boldsymbol{p}^i$ is the position vector of the i-th cell and $\boldsymbol{p}^{i-1}$ and $\boldsymbol{p}^{i+1}$ are the positions of the neighboring cells at the outer side of the morphology.

$$\boldsymbol{v}_N^i = \boldsymbol{n}^i \cdot \boldsymbol{v}^i, \tag{11}$$

$$\boldsymbol{v}_T^i = \boldsymbol{t}^i \cdot \boldsymbol{v}^i, \tag{12}$$

where $\boldsymbol{v}^i$ is the velocity of the i-th cell.

3 Evolution of Swimming Behaviors

An extended (μ,λ) evolution strategy with individual strategy parameter adaptation has been employed in this work. The strategy parameters (σ) are bounded to $\sigma_m \in [1e^{-6}, 1e^{-4}]$ and $\sigma_c \in [1e^{-6}, \infty]$ for the chromosome for morphological development and that for behavior control, respectively. For the chromosome for morphological development, we also use gene duplication and transposition in addition to mutations. In one setup, gene deletion is also applied, refer to Table 2.

We minimize the following fitness function:

$$f = f_{swim} + f_{shape}, \tag{13}$$

where f_{swim} defines the distances between the center of masses of the body plan at the beginning and the end of the simulation:

$$f_{swim} = -\left| \left(\sum_{i=0}^{n} \boldsymbol{x}^i(t=0) \right) - \left(\sum_{i=0}^{n} \boldsymbol{x}^i(t_{end}) \right) \right| \tag{14}$$

and f_{shape} awards elongated shapes:

$$\begin{aligned} f_{shape} = \max\left\{ \min_i \left\{ \boldsymbol{x}^i(0) \right\}, -30 \right\} - \min\left\{ \max_i \left\{ \boldsymbol{x}^i(0) \right\}, 30 \right\} ... \\ ... - \min\left\{ \min_i \left\{ \boldsymbol{x}^i(1) \right\}, -5 \right\} + \max\left\{ \max_i \left\{ \boldsymbol{x}^i(1) \right\}, 5 \right\}, \end{aligned} \tag{15}$$

where the best reachable value for f_{shape} is -50.

To limit the computational cost, a maximum of 501 cells is allowed. If this number is exceeded before 20 iterations, the developmental process will be stopped. To have a meaningful morphology, the number of cells should be larger than 10. In case that the number of cells is larger than 500 or smaller than 10, a strong penalty is applied.

In the experiments, we set $\mu = 30$ and $\lambda = 200$. The developmental process is computed for $t_{dev} = 20$ iterations. We run the evolution with three slightly different setups, which are listed in Table 2.

Table 2. EA setups

	initial # RUs	initial # SUs	p_{transp}	p_{dup}	p_{del}
Setup 1	66	54	0.05	0.05	0
Setup 2	30	30	0.05	0.05	0
Setup 3	30	30	0.05	0.02	0.03

4 Results and Analysis

In the following, we analyze the fitness profile, the morphology and the control strategy of three animats, picked out from generation 1518, 541 and 197 in the three setups, respectively. We have plotted the fitness for swimming behavior f_{swim} and that for morphology f_{shape} separately every 10 generations in Fig. 3. The f_{shape} of the three animats are -4.21, -50.0, and -9.42, respectively, while the f_{swim} are -97.38, -107.50, and -84.87, respectively. From Fig. 3, we can see that the "fitnesses for shape" in both setups 1 and 3 have converged to a local minimum around generation 150. As a result, the morphology from these two setups is much smaller than that obtained from setup 2, refer to Fig. 4. Another observation is that the fitness for swimming in setup 2 improves steadily in 450 generations and is better than that from setups 1 and 3. Actually, the best animat evolved in setup 2 reached the border of the simulation area within the predefined simulation time.

More interestingly, we found that animats have also evolved different strategies for swimming. In setups 1 and 3, where the evolved morphology is very short, a control period of $T = 10$ is evolved. By contrast, a period $T = 16.3$ has been evolved for the animat in setup 2. In other words, the frequency of the rhythmic

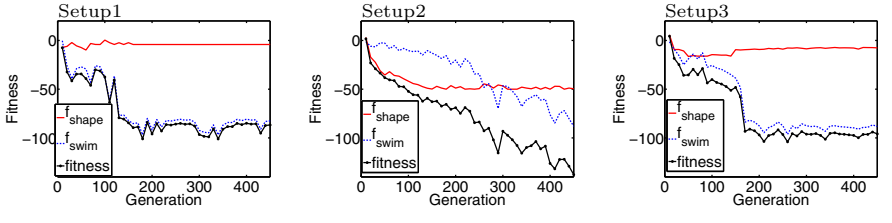


Fig. 3. Fitness profiles from the three setups, which are plotted every 10 generations

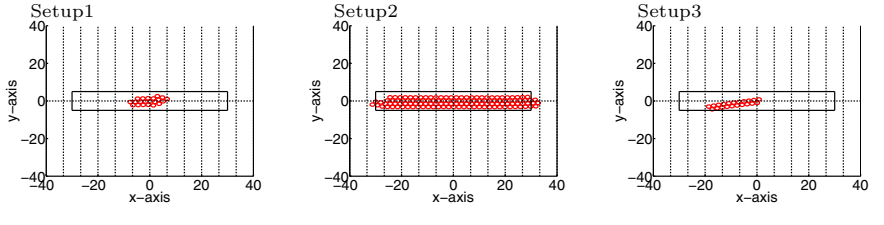


Fig. 4. Morphologies of the three individuals

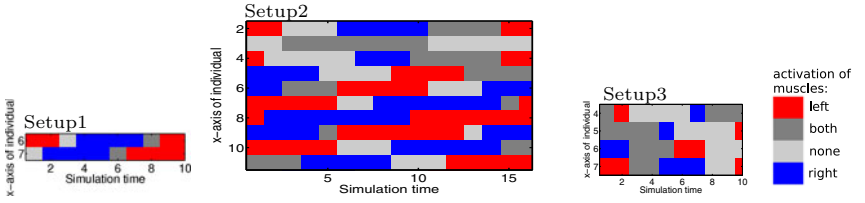


Fig. 5. Evolved phase coordination strategies of the three individuals

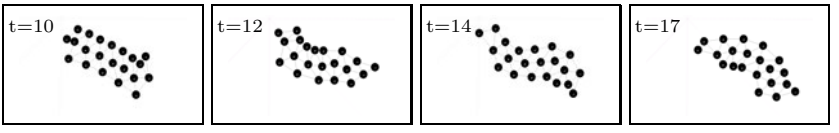


Fig. 6. Snapshots of the movement of the analysed individual from setup 1

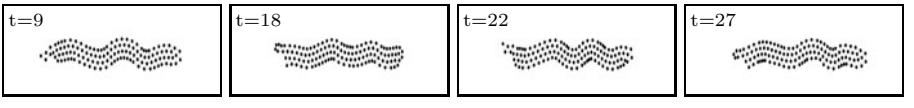


Fig. 7. Snapshots of the movement of the analysed individual from setup 2

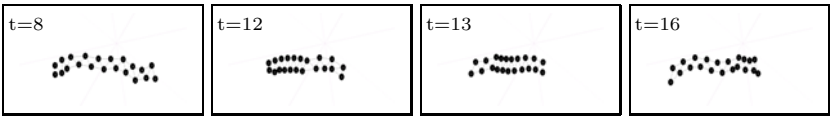


Fig. 8. Snapshots of the movement of the analysed individual from setup 3

movement of the shorter animats is much faster than that of the large one, which makes sense for improving the swimming efficiency. In addition, different phase coordination strategies have also been evolved for the three animats, as shown in Fig. 5. From the phase shift patterns, we can see that animats from setups 1 and 2 produce undulatory movements, while the animat from setup 3 generates peristaltic movements, similar to a caterpillar. A few snapshots of the resulting movement patterns of the three animats are presented in Fig. 6, Fig. 7, and Fig. 8, respectively (videos are available at `www.rtr.tu-darmstadt.de/coevolution`).

5 Conclusion and Future Work

We have presented a model for co-evolving morphological development and motor control for swimming animats. The morphological development is based on a cellular growth model regulated by a GRN, whilst the motor control is represented by the period and phase shifts of the springs. Compared to the direct graph model used by Sims [2] and the L-system by Horn and Pollak [3], our model for morphological development is biologically more plausible. From three

slightly different setups of the evolutionary algorithm, three swimming patterns have emerged, which are adapted to the different morphologies. As far as we know, this is the first work to present a gene-regulated multi-cellular model for morphological development of animats that can perform a functional behavior and to disclose a coupling between the motor control strategy and the body plan.

In future work, we will include a GRN-based model for neural development so that both the neural system and body morphology are subject to a developmental process. By investigating brain-body co-evolution with such biologically plausible models, we hope to gain deeper insights into the co-evolution of nervous systems and morphologies in biology.

Acknowledgments. We would like to thank T. Steiner, M. Olhofer, D. Weiler and N. Einecke for inspiring discussions.

References

1. Taylor, T., Massey, C.: Recent developments in the evolution of morphologies and controllers for physically simulated creatures. Artificial Life 7(1), 77–87 (2001)
2. Sims, K.: Evolving virtual creatures. In: Proceedings SIGGRAPH, pp. 15–22 (1994)
3. Hornby, G., Pollack, J.: Evolving L-systems to generate virtual creatures. Computers & Graphics 25, 1040–1048 (2001)
4. Miconi, T., Channon, A.: An improved system for artificial creatures evolution. In: Proceedings of Artificial Life X, pp. 255–261 (2006)
5. Spector, L., Klein, J., Feinstein, M.: Division blocks and the open-ended evolution of development, form, and behavior. In: Proceedings of GECCO, pp. 316–323 (2007)
6. Kitano, H.: A simple model of neurogenesis and cell differentiation based on evolutionary large-scale chaos. Artificial Life 2(1), 79–99 (1995)
7. Eggenberger, P.: Evolving morphologies of simulated 3d organisms based on differential gene expression. In: Proceedings of Artificial Life IV, pp. 205–213 (1997)
8. Bongard, J.C., Paul, C.: Investigating morphological symmetry and locomotive efficiency using virtual embodied evolution. In: Proc. of SAB, pp. 420–429 (2000)
9. Jones, B., Jin, Y., Sendhoff, B., Yao, X.: Evolving functional symmetry in a three dimensional model of an elongated organism. In: Artificial Life XI, pp. 305–312 (2008)
10. Mazzapioda, M., Cangelosi, A., Nolfi, D.: Co-evolving morphology and control: A distributed approach. In: Congress on Evolutionary Computation (2009) (in press)
11. Oros, N., Steuber, V., Davey, N., Canamero, L., Adams, R.: Evolution of bilateral symmetry in agents controlled by spiking neural networks. In: IEEE Symposium on Artificial Life, pp. 116–123. IEEE Press, Los Alamitos (2009)
12. Jones, B., Jin, Y., Yao, X., Sendhoff, B.: Evolution of neural organization in a hydra-like animat. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008. LNCS, vol. 5506, pp. 216–223. Springer, Heidelberg (2009)
13. Steiner, T., Schramm, L., Jin, Y., Sendhoff, B.: Emergence of feedback in artificial gene regulatory networks. In: Proc. of CEC, pp. 867–874 (2007)
14. Jin, Y., Schramm, L., Sendhoff, B.: A gene regulatory model for the development of primitive nervous systems. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008. LNCS, vol. 5506, pp. 48–55. Springer, Heidelberg (2009)
15. Klein, J.: BREVE simulation environment. Open-Source Software Package, Source Code Available at, http://www.spiderland.org/
16. Sfakiotakis, M., Tsakiris, D.: Simuun: A simulation environment for undulatory locomotion. International Journal of Modelling and Simulation 26(4), 350–358 (2006)

Evolution of the Morphology and Patterning of Artificial Embryos: Scaling the Tricolour Problem to the Third Dimension

Michał Joachimczak and Borys Wróbel

Computational Biology Group, Department of Genetics and Marine Biotechnology
Institute of Oceanology, Polish Academy of Sciences
Powstańców Warszawy 55, 81-712 Sopot, Poland
{mjoach,bwrobel}@iopan.gda.pl

Abstract. We present a model of three-dimensional artificial embryogenesis in which a multicellular embryo develops controlled by a continuous regulatory network encoded in a linear genome. Development takes place in a continuous space, with spherical cells of variable size, and is controlled by simulated physics. We apply a genetic algorithm to the problem of the simultaneous evolution of morphology and patterning into colour stripes and demonstrate how the system achieves the task by exploiting physical forces and using self-generated morphogen gradients. We observe a high degree of robustness to damage in evolved individuals and explore the limits of the system using more complex variations of the problem. We find that the system remains highly evolvable despite the increased complexity of three-dimensional space and the flexible coding of the genome requiring from evolution to invent all necessary morphogens and transcription factors.

Keywords: artificial embryogeny, gene regulatory network, morphogenesis, embryo patterning, positional gradients, French flag model, cellular differentiation.

1 Introduction

The field of artificial embryogenesis investigates how the compact genomes of living organisms are able to encode the structure of extremely complex multicellular organisms using a limited number of genes. As a rule, multicellular biological organisms start their development from a single cell and form through cell division. Before each division, a copy of the genome is made. Symmetry-breaking mechanisms cause the differential expression of the genes in the descendant cells. However, the embryo is shaped not only by the fact that different gene products are present at different concentrations in different cells. Physical interactions between the cell components, as well as between the cells and their environment result in emergent phenomena that allow bridging the overwhelming discrepancy between the amount of information encoded in the genome and the complex structure of the organism. Furthermore, living developing systems display a remarkable robustness to perturbations. On the one hand, this demands better understanding, while on the other, it raises hopes of employing developmental models

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 35–43, 2011.

to solve various engineering problems and to overcome the limitations of evolvability present in direct encoding schemes ([1,2]).

Many approaches to generating morphologies of artificial organisms have been proposed, and have frequently been evaluated using the so-called French flag problem, proposed more than 40 years ago by Lewis Wolpert ([3]), which consist of developing a patterned embryo with three colours (blue, white, red) in three areas. Several recent papers present biologically-inspired models of cellular development in which each cell is controlled by a gene regulatory network. In some, the development takes place on a 2D lattice with cells occupying fixed locations (e.g., [4,5,6]). In others, for example those using the Cellular Potts Model, the cells occupy multiple locations on a lattice, and their shapes are determined by the simulated physical interactions of their membranes ([7,8]). Other approaches include elastic interactions between the cells ([9] and recently [10]). Three-dimensional simulations ([11,9,12,13]) remain scarce, partly because it is not always necessary to investigate underlying phenomena, and partly because more degrees of freedom make 3D multicellular development a much more difficult task.

In this work, we attempt to scale the French tricolour problem to three dimensions by evolving genomes which code gene networks regulating the development of multicellular ellipsoidal bodies with multicolour patterning. Patterning (cell differentiation) is understood as expressing particular gene products in different areas of the embryo. The cells in our model form bodies through elastic interactions, are suspended in continuous space (no grid is used), and produce spatiotemporal gradients of morphogens, perceived by other cells.

2 The Model

We employ an extended version of the system introduced in [12], where it was used to evolve three-dimensional morphologies of multicellular organisms. The conceptual framework remains unchanged, but many features that were found to be superfluous were removed, facilitating a better understanding of and control over the whole system. Furthermore, biological relevance was improved by modelling product accumulation and degradation. An overview of essential features and concepts is provided, for additional details please refer to [12]. As the main interest of our work is the emergent properties of evolved regulatory networks, our system is designed to minimize limitations imposed on topologies that can evolve. In our model, linear genomes can contain any number of regulatory units. Regulatory units are composed of one or several regulatory elements and one of several genes (genetic elements that code products, which include transcription factors and morphogens). Any number of connections between regulatory units can exist.

2.1 Genome and Genetic Elements

The genome in our model is a list of genetic elements that fall into three classes: elements that code products (called genes); regulatory elements (called promoters); special elements (that code the external inputs and outputs of the regulatory network). The genome is parsed sequentially, and regulatory units are formed whenever a series of

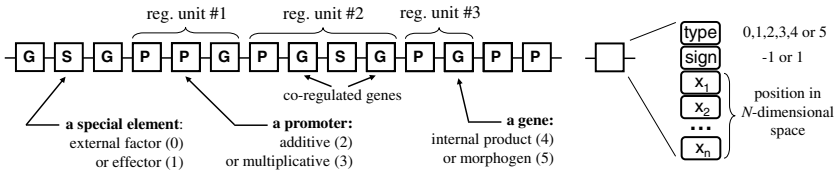


Fig. 1. The genome and the structure of a single genetic element. Each element consists of a type field, a sign field, and a sequence of N real values used to determine affinity to other elements ($N = 2$ was used in this paper).

promoter elements is followed by a series of genes, with special elements assigned to input and output nodes at a later stage. By computing affinities between all gene products and all promoters, connections between regulatory units are formed, and, thus, a gene regulatory network (GRN) emerges. Fig. 1 provides an overview of the process. In our system, promoters come in two types: additive and multiplicative (see below). The products can be either internal or external; only external products (morphogens) can diffuse from one cell to another.

Each genetic element in our system encodes a point in N-dimensional space (Fig. 1). Affinity between products and promoters is determined as a function of Euclidean distances between associated points in this space. The lower the Euclidean distance is between these points, the higher the affinity is between gene products and promoters. A cut-off distance is used to prevent full connectivity in the network. The product of sign fields of the two elements determines the sign of the connection (activatory or inhibitory). One can imagine (or actually visualize if $N < 3$) that as genomes evolve (and the element coordinates change), points in N-dimensional sequence space that correspond to the elements approach one another or move away. Note, however, that this space is used solely as a mechanism for determining connectivity and bears no relation to the 3D space in which multicellular development takes place.

The activation of each promoter in a regulatory unit is the sum of the concentration of all binding products weighted by their affinities. The sum of activations of all additive promoters is multiplied by the product of activation of all multiplicative promoters. The result (A) is fed to a sigmoid function $f_A(A) = \frac{2}{1+e^{-(A-1)}}$. This is interpreted as the production rate (positive or negative) of all products in a given regulatory unit, i.e., $\frac{dL}{dt} = f_A(A) - L$, where L is the current concentration. This results either in an increase of concentration of a product (if the synthesis is higher than degradation) or in increased degradation. The presence of a multiplicative promoter in a regulatory unit results in a strict requirement for the presence of a binding product, otherwise the unit is not expressed.

2.2 Development

Cells occupy continuous positions in 3D space and are modelled as elastic spheres (of various sizes). If two cells overlap, a repulsive force proportional to the amount of overlap is applied. In particular, after cell division the daughter cell overlaps with the mother and will slowly move away. To maintain the coherent structure of the embryo, adhesive forces are simulated: cells stick together whenever they are at a close distance.

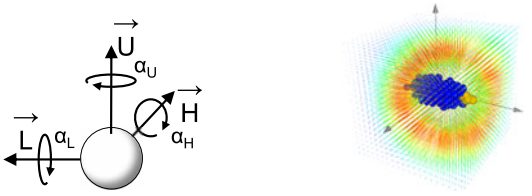


Fig. 2. Mechanics of the developmental model. Left: each cell has an internal orientation vector $\boldsymbol{H}$ and two perpendicular helper vectors that can be rotated by three effectors (outputs of the network) - see p. 19 of [14] for explanation of such a system. Right: simplified diffusion is modelled without the use of a grid, by using past values of morphogen expressions. An outburst of morphogen production in the embryo followed by quick degradation produces a wave travelling through the system.

Furthermore, fluid drag is simulated to prevent erratic movements. Morphogen diffusion (Fig. 2, right) is simulated by calculating the perceived levels of diffusive substances in a given point in space as a function of distance and the historic values of production from all the sources (the farther the source, the older value is used).

Special elements code either for the outputs of the network (effectors) or inputs. Effectors correspond to specific cell action (division, death, size increase of the daughter cell, and rotation of the cell orientation vector; cell orientation determines where the new cell is placed after division, similarly to 3D L-systems [14]). In addition, the composite of the activation of colour effectors defines the cell colour. We use either two (red, blue) or three (red, green, blue) colour effectors, so for example, cells that express both red and blue are pink. Division or death of a cell occurs when the activation of the corresponding effector is above a set threshold. Similarly, in some experiments cells were coloured only when the activation of the colour effector was above a set threshold.

In all the experiments in this work, the GRNs are given as the input an external factor that was present in all cells at the same concentration (set at 1) throughout development. This input was necessary to initiate any activity, with the exception of experiments where two maternal morphogen gradients were deployed in the environment and could serve the same purpose. Direct regulatory connections between external factors and effectors were not allowed to prevent trivial solutions.

At the time of division, the state of GRN is copied to the new cell, which can differ from the mother only by its rotated orientation vector and size. As these properties cannot directly influence the state of GRN, an external symmetry breaking mechanism is necessary. Development is allowed to take 300 time steps, in each time step both the position of the cells and the state of GRN in each cell is updated.

2.3 Genetic Algorithm and the Fitness Function

Genetic operators in our system act on the level of element fields (changing element type, sign bit, or disturbing the coordinates of an associated point in space). Duplications and deletions of single elements or multiple elements are allowed. The results shown in this work were obtained using a fairly standard genetic algorithm with a

population size of 300, elitism, tournament selection, and multipoint crossover for sexual reproduction (for 30% of the individuals in each generation). Evolutionary runs were initiated with individuals consisting of 5 randomly created regulatory units, usually requiring a few thousand tries before a single individual capable of starting division would appear (as this requires random emergence of connection from external factor '1' to some regulatory unit and then to the division effector). Evolutionary runs where terminated after no improvement over 500 generations was detected, resulting in runs lasting for about 2000-3000 generations.

To assess the fitness of individuals, evolved morphology was compared to a target shape at the voxel level. Each voxel outside the target and inside some cell decreased the fitness. Each voxel inside the target and inside a cell increased fitness proportionally to the correctness of colours expressed by this cell (i.e., 1 for perfect match, 0 for completely reversed pattern expression). Thus, both the correct morphology and patterning were rewarded. To reward the emergence of multiple colours at the very early stages of evolution, fitness was divided by the number of colour effectors minus the number of colours present in the individual (as in [8]). We found this improved evolvability greatly.

3 Results and Discussion

We investigated the evolvability and scalability of the system by performing a series of experiments, each repeated 10 times. In the first series of experiments, two colour effectors were used, allowing the cells to be coloured red or blue if their activation (a real value between 0 and 1) was above a threshold of 0.5, resulting in four possible colours - with pink for expression of both effectors and white for no expression (rather than black, so that actual French tricolour could be obtained). The target shape seen in Fig. 3 (left) was used. Development was terminated after 300 time steps, with a hard limit on the number of cells (i.e., cells could not divide after the limit was hit). Obtaining correct morphology turned out to be a fairly trivial task for the GA, owing to the exploitation of physics. Embryos would apply minor variations to the orientation of division vectors and the repulsion of cells would take it from there, creating elongated morphologies (Fig. 3, right).

The only input to the network in these experiments was a static external factor, perceived in each cell at the constant level of '1'. The symmetry of the embryo development is broken because, after division, cells shift in space and perceive slightly different concentrations of morphogens. This results in self-emerging positional information (explored earlier by Knabe et. al. in [8]).

Fig. 3. Left: default target shape and colour pattern,right: small variations to orientations of divisions generate an elongated shape by exploiting the physics of the system (same individual as in Fig. 4)

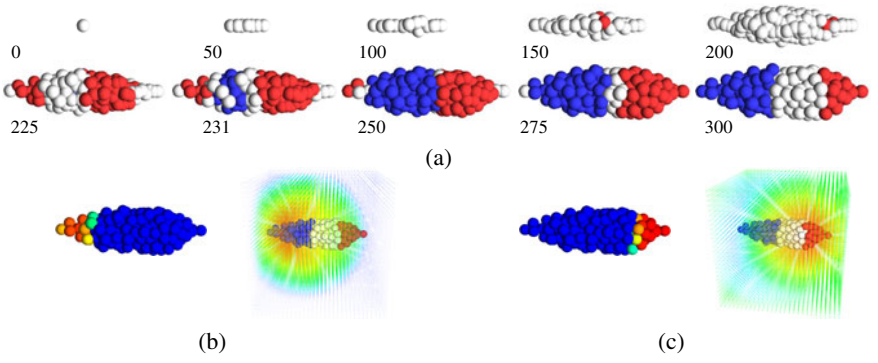


Fig. 4. Development of the best individual with 2 colour effectors (a); below, self-generated gradients of positional information employing two different morphogens (bc): left - production level of a morphogen in each cell (blue to red colour map), right - normalized morphogen density maps in the space surrounding the embryo.

Fig. 4a presents snapshots from the development[1] of the best individual obtained in 10 runs, with a 200 cell limit. Typically, runs would result in 30-40% yield of three-coloured individuals, the others would remain stuck in local suboptima. A more detailed analysis of the individual shown in Fig. 4a allows locating the clear asymmetric production of two distinct morphogens centred in the posterior and anterior of the embryo and reveals gradients generated in space (Fig. 4bc). Only two gradients are shown in the figure, even though two more morphogens with similar patterns were also present. Although this is not the only possible solution for generating anterior/posterior axis (single morphogen at one extreme of the embryo would suffice in principle), all analysed individuals developed using the differential production of at least two morphogens.

One of the problems identified in early experiments was that when evaluated by their similarity with target pattern after 300 steps, the patterns were rarely stable. Typically, they would sweep through the embryo (driven by diffusing waves of morphogens) or oscillate. We have managed to partially alleviate this by calculating overall fitness as an average of similarity values taken every 5 simulation steps in the last 50. Individuals that were largely stable through this period would then be obtained. However, in most cases, the pattern degraded if development was allowed to continue beyond its default lifetime of 300 steps. We note, however, that it is (sadly) a common feature of living systems to degrade if their lifespan is extended beyond what they were selected for by evolution.

In the next series of experiments, we investigated the stabilizing role of the gradients of substances present in the environment of the developing embryo. Two external factors diffused from sources external to the embryo at its two extremes (similarly to Bicoid and Nanos, which determine anterior/posterior axis in Drosophila embryo development; see, e.g., [15]). This allowed for stable embryo patterning, but only if the cells were additionally prevented from producing their own morphogens (which makes

[1] Supplementary materials, including videos of development and gradient formation, are available at http://www.iopan.gda.pl/molbiol/ecal09patterning

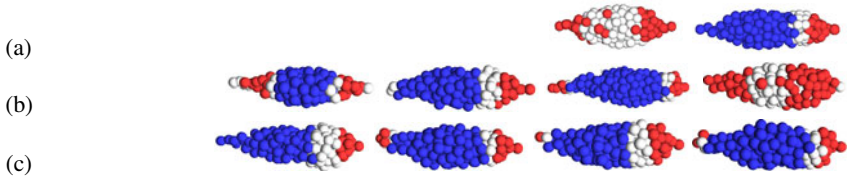


Fig. 5. Robustness to damage: the effects of removing a single cell from the embryo at the 2- (a), 4- (b), or 8-cell stages (c) of development; in (c) only half of possible cases is shown. Removing cells at further stages has a notable, albeit diminishing, effect. The same individual as in Fig. 4 is shown.

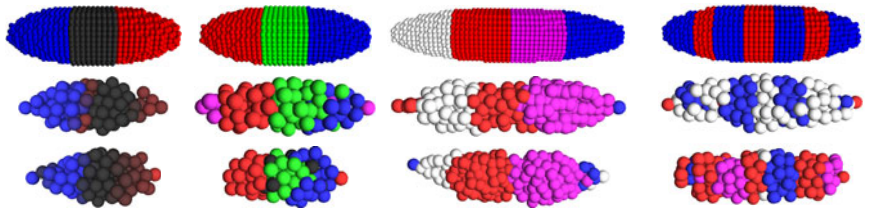


Fig. 6. More difficult evolutionary targets presented to the system (top); two best individuals out of 10 runs for each target are shown below. Note that for (a) and (b) the lack of expression of colour effectors is drawn as black.

those maternal factors the only inducers of differential expression). The dominant role of self-produced morphogens can be explained by the fact that, in our system, morphogens produced by the embryo can reach much higher concentrations than the diffusive substances present in the environment.

Robustness to perturbations such as mutations or damage is one of the essential features observed in developmental systems (see e.g. [2,4,16]). Although all the individuals presented in this paper evolved in the absence of any developmental stochasticity, they were found to respond extremely well to the removal of a single cell, even at the very early stages of development (Fig. 5). This suggests that fault tolerance developed as an effect of mutational robustness and can be considered to be an indication of good evolvability ([17]).

To assess the scalability of the model, different approaches to patterning were evaluated. Fig. 6a shows the use of non-thresholded colour effectors. This turned out to be a harder problem, which is understandable if one considers that it was now not only necessary to reach a certain threshold of colour expression, but also to maximize (or repress) it in a given section of the embryo. Rather surprisingly, evolutionary runs (terminated after 500 generations without improvement) took much longer, taking even up to 30 000 generations (compared to about 3 000 in the experiments with thresholding). This indicates that the fitness landscape of this problem is actually much less rugged, and provides many more opportunities to fine-tune individuals.

The task can also be made harder by introducing the third colour effector (Fig. 6b), making it necessary to both express a certain colour and repress the expression of two others in each area. As expected, lower fitness was reached in general, but some tri-coloured individuals were obtained. Fig. 6cd show solutions to other variations of the patterning problem - the emergence of four areas or multiple stripes. Both tasks were solved using two colour effectors. These problems are still within range but approaching the limits of evolvability of the current setup.

4 Conclusions

We present a system capable of evolving solutions to the French flag problem using simulated physics and self-generated gradients of morphogens dynamically diffusing in the environment. The computational cost of adding a third dimension and implementing simple physics was negligible compared to the cost of the simulation of a regulatory network in every cell. Considering the availability of sophisticated physics engines, it would be very interesting to see how this developmental system (and other comparable models) would be able to exploit more complex simulated physics, such as flexible cells. Thus, our further work will focus on maintaining the biologically-relevant features of the network while introducing more elements to simulate the physical aspects of multicellular development in our system.

Acknowledgements. The computational resources used in this work were obtained through the support of the Polish Ministry of Science and Education (project N N303 291234 and N N519 384236), the Tri-city Academic Computer Centre (TASK) and the Interdisciplinary Centre for Molecular and Mathematical Modelling (ICM, University of Warsaw; project G33-8).

References

1. Bentley, P., Kumar, S.: Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In: Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, USA, vol. 1, pp. 35–43. Morgan Kaufmann, San Francisco (1999)
2. Stanley, K., Miikkulainen, R.: A taxonomy for artificial embryogeny. Artificial Life 9(2), 93–130 (2003)
3. Wolpert, L.: Positional information and the spatial pattern of cellular differentiation. J. Theor. Biol. 25(1), 1–47 (1969)
4. Miller, J.F.: Evolving a self-repairing, self-regulating, french flag organism. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 129–139. Springer, Heidelberg (2004)
5. Beurier, G., Michel, F., Ferber, J.: A morphogenesis model for multiagent embryogeny. In: Proceedings of ALIFE X, pp. 84–90. MIT Press, Cambridge (2006)
6. Chavoya, A., Duthen, Y.: A cell pattern generation model based on an extended artificial regulatory network. Biosystems 94(1-2), 95–101 (2008)
7. Hogeweg, P.: Evolving mechanisms of morphogenesis: on the interplay between differential adhesion and cell differentiation. J. Theor. Biol. 203(4), 317–333 (2000)

8. Knabe, J.F., Nehaniv, C.L., Schilstra, M.J.: Evolution and morphogenesis of differentiated multicellular organisms: autonomously generated diffusion gradients for positional information. In: Proceedings of ALIFE XI, pp. 321–328. MIT Press, Cambridge (2008)
9. Hotz, P.E.: Genome-physics interaction as a new concept to reduce the number of genetic parameters in artificial evolution. In: Congress on Evolutionary Computation, CEC 2003., vol. 1, pp. 191–198 (2003)
10. Doursat, R.: Programmable architectures that are complex and self-organized: From morphogenesis to engineering. In: Proceedings of Artificial XI, pp. 181–188. MIT Press, Cambridge (2008)
11. Kumar, S.: Investigating Computational Models of Development for the Construction of Shape and Form. PhD thesis, Dept. of Computer Science, University College London (2004)
12. Joachimczak, M., Wróbel, B.: Evo-devo in silico: a model of a gene network regulating multicellular development in 3D space with artificial physics. In: Proceedings of Artificial XI, pp. 297–304. MIT Press, Cambridge (2008)
13. Haddow, P.C., Hoye, J.: Achieving a simple development model for 3D shapes: are chemicals necessary? In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO 2007), pp. 1013–1020. ACM, New York (2007)
14. Prusinkiewicz, P., Lindenmayer, A.: The algorithmic beauty of plants. Springer, New York (1996)
15. Carroll, S.B., Grenier, J.K., Weatherbee, S.D.: From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design. Blackwell Publishers, Malden (2004)
16. Eggenberger, P.: Exploring regenerative mechanisms found in flatworms by artificial evolutionary techniques using genetic regulatory networks. In: The Congress on Evolutionary Computation, CEC 2003, vol. 3, pp. 2026–2033 (2003)
17. Federici, D., Ziemke, T.: Why are evolved developing organisms also fault-tolerant? In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D. (eds.) SAB 2006. LNCS (LNAI), vol. 4095, pp. 449–460. Springer, Heidelberg (2006)

Artificial Cells for Information Processing: Iris Classification

Enrique Fernandez-Blanco, Julian Dorado, Jose A. Serantes, Daniel Rivero, and Juan R. Rabuñal

University of A Coruña, Campus Elviña, A Coruña, Spain
{efernandez,julian,jserantesv,drivero,juanra}@udc.es

Abstract. This paper presents a model in the Artificial Embryogene (AE) framework. The presented system tries to model the main functions of the biological cell model. The main part of this paper describes the Gene Regulatory Network (GRN) model, which has a similar processing information capacity as Boole's Algebra. This paper also describes how to use it to perform the Iris Classification problem which is a pattern classification problem. The aim of this work is to show that the model can solve this kind of problems.

Keywords: Artificial Embryogeny, Genetic Algorithms.

1 Introduction

Nature presents a lot of different systems that can be used in Computer Science as an inspiration to develop new tools. Examples like Artificial Neural Networks (ANNs) or Genetic Algorithms are well-known. This work starts from the idea that any cell of a biological tissue has to communicate and process the signals of its environment. This behavior can be seen as distributed computation, where each cell plays the character of a single processor and it has to coordinate its computation with its neighbor cells. Nature just needs a few signals from the environment and the information contained in the DNA to develop and coordinate the most complicated structures.

The objective of the present work is to develop a model inspired in embryological cells, which have features like self-organization, self-reparation, etc. To develop the computer adaptation, the biological model was simplified by identifying and modeling the essential elements. In this way, some parts of the computer adaptation have very similar functions to the biological ones (DNA, gene or cytoplasm, etc.). The main objective of this paper is to adapt this model in order to apply it to information processing problems and, in particular, to classification and pattern recognition problems.

2 Background

In 2003, Stanley and Miikmulainen [1] developed a methodology to classify the different AE models that appear in Evolutionary Computation (EC). These models are

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 44–52, 2011.

based on abstractions of the embryological cells, which can be classified into two main types. On one hand, some works follow a grammatical approach, where the most important works are related to L-systems [2]. On the other hand, other studies have a chemical-oriented approach based on Turing's ideas [3].

The works related to the grammatical approach have been mostly used to develop ANN. Kitano's work [4] shows how the connectivity matrix of an ANN is evolved with a set of rules. Another remarkable work is [5], in which the authors develop both the control and the structure of the robot using L-systems.

For the chemical approach, the first work to be mentioned is [6], in which Kauffmann develops his Gene Regulatory Networks [6]. The objective of the works which follow this approach is usually its application to different problems, such as approximating a simple figure/structure in a 3D space, or the development of evolutionary hardware [7]. One of the most important works which tries to solve the previously mentioned problems is described in [8]. The most interesting part is the usege of fractal proteins for the communication among the cells of the model.

The model presented in the current paper can be included into the chemical approach. The model has included most of the concepts present in the previously mentioned works, like the concept of operon or the cellular division and death. The most novel concept is that never before a chemical approach has been used to solve an information processing problem.

3 Model

Every cell of any biological tissue has as antecessor: a unique cell, called zygote, which generates other cells and they can coordinate their behavior using the information present in DNA. Each cell knows its purpose from its DNA and the proteins that it receives. Therefore, it can be considered that each cell of the tissue works as a processor and all of them operate with proteins using the DNA as operator set. Self-reparation, self-organizationand parallel information processing are some features of the structures generated with this computation model [9].

Below, the structures of the artificial model that arise from the study of the biological issues are explained.

Protein
Protein is the basic piece of information. In this model, proteins are a string of bits that identifies each one of the different proteins and has a time to live (TTL). Due to this, the system has a memory of previous generated proteins, until they are used or degraded.

Cytoplasm
Cytoplasm is the part of the artificial cell which has the responsibility of managing the information inside the cell. The responsibility of this part is to manage the proteins needed for a transcription in the cell and to check the concentration level of the proteins inside and outside the cell to decide which proteins will be communicated.

Gene
Each gene of the system represents a rule, where some conditions have to be fulfilled to perform a certain computation or process. The genes are strings of bits which contain two parts: promoters and a gene identifier.

- **Promoter region**. This part identifies the proteins needed to activate this gene. This section can appear several times and it is composed of two parts:
  - **Promoter Sequence**. This section identifies the required protein sequences to activate the gene.
  - **Concentration lock**. Each of the activation proteins needs a certain concentration level identified in this field in order to activate this gene.
- **Gene identifier**. This part identifies which is the protein generated by the gene and the type of the gene. It is composed of two subparts.
  - **Generated sequence**. When the gene is activated, the result of that activation is a protein with this sequence.
  - **Constitutive mark**. This bit indicates if the gene is a constitutive gene (explained below).

The required proteins have to be at least in a certain proportion inside the cell. This minimum concentration is stored in its concentration lock part. The protein does not need to be identical to the activation protein. As in Nature, high concentrations of similar proteins can activate a gene. This fact is modeled using the following condition:

$$\text{Protein Concentration Percentage} >= (\text{Distance} + 1) * \text{Concentration Lock} \quad (1)$$

In the previous condition, *Protein Concentration Percentage* represents the concentration of a protein inside the cell. *Distance* is the hamming distance between that protein and the activation protein. Finally, *Concentration Lock* is the concentration required for the promoter sequence. If the condition is met for all of the promoters, then the gene generates a protein with the gene's generated sequence.

This is the normal behavior of a gene but when it is marked as constitutive its behaviour changes drastically. A constitutive gene is constantly generating proteins, until its activation proteins appear. In this case, these are called inhibitor proteins. When the condition Eq. 1 is met by the inhibitor proteins, the gene stops producing proteins during a certain period.

Operon

Operon is a group of genes which codify a task. In Nature, these genes codify the most complex tasks and act all or none of them. This idea was adapted by a structure which applies conditions to a group of genes. This structure has the same parts as a gene and acts in the same way but, instead of a generated sequence, it has a group of genes. This allows the activation of those genes from that moment for a period of time.

DNA

The DNA is composed of genes and operons. Its responsibility is to select the allowed genes which can be activated, when the cell asks for them in a certain moment of its development.

Cell

Cells are the basic element of the system and contain all the previously described parts (DNA, Gene, etc.). The expression of the DNA can induce the cell to communicate with its environment. The expression of the DNA can also induce two special behaviors: division or death of the cell.

Cells define a concept to regulate the different actions, which is the cellular time or *cellular cycles.* These *cellular cycles* contain all the tasks that a cell can do at the same time. For example, a cell can communicate proteins to the environment many times, but it can only divide itself once in a cellular cycle. The steps of a cycle are the following:

1. Update the concentration of the proteins and delete those whose TTL reaches zero.
2. Process the DNA with the cytoplasm proteins to generate new proteins.
3. Check the concentration of the proteins and execute the communications of the cell, or execute one of the special actions (divide and death) if it is necessary.

In short, after the actualization of the proteins' TTL, when a cell activates a gene, four different actions can be performed: storage the protein in the cytoplasm; communicationamong cells the proteins; division of the cell or death of the cell. All of these actions are performed by each cell in such a way that the cellular cycle is used to coordinate the actions.

Environment

The environment is where the cells are and it determines the type of communication among cells. The main purpose of the environment is to manage the free proteins. The free proteins are those which have been set out by a cell and no cell has already required them. The environment determines how these proteins move until they are required by a cell or when they are deleted because their TTL has expired.

Communication Model

Once the parts and the environment have been defined, the next step is to define how these elements interact among them. Each cell can communicate with its neighborhood by using the proteins. Proteins are set into the environment when their concentration inside the cell is higher than a certain threshold.

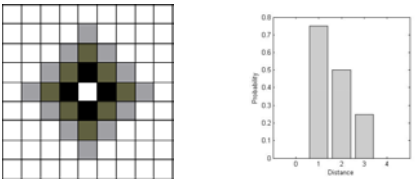


Fig. 1. Reception Probability of a Protein

The proteins set into the environment modify the return value of a probability function associated to its type. This function gives the probability of finding a protein in a point far away from the emission point. When a new protein is set into the environment, the probability associated to that point is increased. This function has two parameters: the number of proteins in the environment's position and the distance between that point and the checked point. The probability of finding a protein decreases with the distance (Fig. 1). The decreasing value is represented in Fig. 1 by the darkness in the squares that surrounds the emission point. Cells check in each *cellular cycle,* the proteins that have any possibility to be caught by them. If the protein is taken by a cell, then the value of the function is decreased to represent the reduction of the amount.

4 Instruction Search Method

To search the instructions with the shape of an artificial DNA strand, a Genetic Algorithm (GA) has been used [10]. The reasons for choosing a GA are: it is one of the most robust and adaptable methods and the features of a GA match the dynamics of the presented system.
The GA's DNA is not a fixed-length array of variables because the number of rules and the number conditions of these rules (promoter sequence) are unknown This paper proposes the subdivision of the DNA in functional sections, (see Fig. 2):

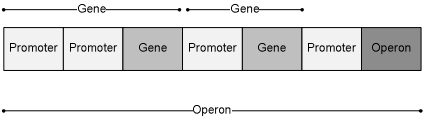


Fig. 2. Compiling a DNA

- **Promoter** sections are sections that codify a gene's promoter region.
- **Gene** sections identify a gene of the cellular system and codify the gene identifier. This section is associated to the previous promoter sections that appear in the GA individual until another Gene identification section or Operon section is found.
- **Operon** section is a mark of the creation of an operon of the cellular system. This operon has as promoters the previous promoter sections, until a Gene or operon identification section appears. The genes contained into the operon are those which have been identified before the operon. The operon contains the genes, until a maximum number of genes is reached, or it finds another operon identification section is found.

These sections and the association show in Fig. 2 are able to search all the possible combinations of genes. To simplify the GA's functioning, the sections used by the GA have all the fields of the three functional sections, but these sections have two extra bits to determine the type and which are the valid fields. The next step is to adapt the GA crossover and mutation operators to work with the variable length.

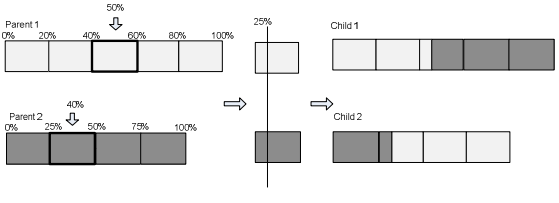


Fig. 3. Crossover Example

The crossover operator used is a one point crossover, which selects the crossover section according to the length of each individual. A random percentage over the length, which is the number of functional sections, is generated for each individual

and the section in that position is selected. Both parents randomly choose the same point inside the sections to make the crossover and the children are the combination of the parents' genetic material as shows Fig. 3.

The mutation operator can execute one of these three possible operations:

- Add a new section to an individual.
- Delete a section.
- Change a bit inside a section.

In the mutation operator, the operation which changes a bit needs more explanation because each section has two bits that identify its type at the beginning of the binary string. The first bit identifies if it is a promoter section or not. The second bit determines if it is an operon or a gene (when it is not a promoter). A change in any of these bits makes the information of the section to be reinterpreted in its new role because the new role activates different fields inside the section used by the GA. Other changes in bits only change the associated value of the field. Note that the GA sections have all the fields of the functional sections to simplify this step.

Finally, the probabilities of executing each of the actions, empirically obtained, are: Addition (20%), deletion(20%) and change (60%). The percentage is the probability of executing each operation each time a mutation is executed.

5 Test

This DNA model defines a Gene Regulatory Network (GRN). This kind of structure has proved to have the same capacities as Bool's Algebra [6].

To adapt the model to process information problems, the inputs and outputs of the system have to be defined. As inputs, some source points put in the environment new proteins each *cellular cycle*. The outputs are measured by setting some sinks which get the proteins from the environment. The final element of the system is the set of cells into the environment, which process the inputs and set the outputs. The objective of the test is to search DNA which minimizes the output error in the sinks when the inputs are present in the source points. The DNA is put into a cell and the input values are put in the source points. After a number of cellular cycles, the output values are checked in the sink points.

The iris classification problem is a non-linear classification problem. Four parameters were measured in millimeters on 50 iris specimens from each of three iris species, *Iris setosa*, *Iris versicolor* and *Iris virginica [11]*. Given the four parameters, one should be able to determine which of the three classes a specimen belongs to. The patterns are distributed in three areas, one for each class. One of the types presents a clear difference with the other two, but the other two classes are very close and they present a conflict area with no clear differentiation.

$$fitness = \sum_i \begin{cases} 100 & \text{no protein registered} \\ 1 & \text{no desired protein} \\ 0 & \text{desired protein in the first position of the list} \\ 0.5 & \text{desired protein in the second position of the list} \\ 0.75 & \text{desired protein in the third position of the list} \\ 0.85 & \text{desired protein in other position in the list} \end{cases} \quad (2)$$

The model tries to find a DNA which classifies the different patterns of iris flowers. To adapt the model to this problem, the values of the variables were normalized between 0 and 1. One sink and 4 source points, one for each variable, were used. The source points put into the environment 4 different proteins, one for each variable. The normalized value of a variable is the peek probability of receiving it around the source point. This probability decreases linearly from the source point to the surrounding area, and it determines the probability to find a protein in that point. Three sequences are identified as the desired outputs and they determine the predicted class of the input data: Iris Setosa (1111), Iris Versicolor (1001) and Iris Virginica (1010). These desired sequences will be the principal component of the fitness function. The GA will seach a DNA which maximizes the explresion of this desired sequences. The ordered list of received proteins is needed to check the position of the desired sequence. The function will add a different penalization depending on the position into the ordered list as shown in Eq. 2. The penalization was selected empirically.

The letter i represents a pattern of the training set. The fitness is the addition of the penalizations for each pattern. These penalization values are determined by the position on the list. There are two special cases: when the desired protein is not present the penalization is 1 and when no protein is received then the error is 100.

Finally, the GA used to search the DNA, which allows this classification, has shown a better behavior with a 70% crossover rate and a 30% mutation probability in populations with a size between 50 and 100 individuals.

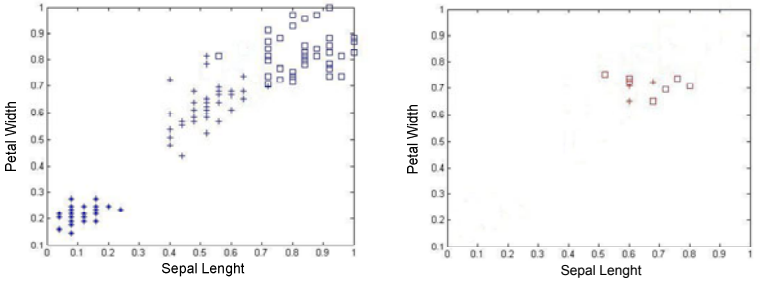


Fig. 4. Iris Flower correct classification. Correctly classified (left), wrongly classified (right).

The best individual has a fitness of 8.85 and it is able to classify 140 of 150 patterns. Fig 4 shows the three kinds of iris flowers represented by a star, a cross and a square. These results were obtained using only one cell to solve the problem. More cells would increase the complexity of the solution, but the resulted structure may classify better the patterns of the conflicted area by the cooperation among cells.

6 Conclusions

This paper presents a model which follows a different approach to previous works. It has two objectives. The first one is to be able to be applied to any kind of problem. The second one is to capture the interesting features of the biological model.

These features, already mentioned in the text, (like self-organization, self-repairation, distributed control, parallel information processing, etc) are those that this model tries to take from Biology and to use in the resolution of problems. The presented system is the base to future developments in the knowledge area of self-organization and distributed computation. The systems built with this cellular model present a high self-organizing and distributed computation level. Future developments with this system could be framed into Autonomic Computing proposed by IBM in 2003[12] because they present many of the features that these systems require. This model could be the base for the communication between different elements in a Autonomic Computing system.

Finally, the tests, in this paper, have shown how the model can be applied to solve different information processing problems. If these results are added to the previous ones [9], a new research area is found. The aim of this area is to develop self-organizing structures which can process information.

Acknowledgement

This work was partially supported by the Spanish Ministry of Education and Culture (Ref TIN2006-13274) and the European Regional Development Funds (ERDF), grant (Ref. PIO52048 and RD07/0067/0005) funded by the Carlos III Health Institute, grant (Ref. PGIDIT 05 SIN 10501PR) and (Ref. PGDIT 07TMT011CT) from the General Directorate of Research of the Xunta de Galicia and grant (File 2006/60, 2007/127 and 2007/144) from the General Directorate of Scientific and Technologic Promotion of the Galician University System of the Xunta de Galicia. Thanks to the Galician Supercomputation Center (CESGA) which has execute a great part of the tests.

References

1. Stanley, K., Miikkulainen, R.: A Taxonomy for Artificial Embryogeny. In: Proceedings of Artificial Life 9, pp. 93–130. MIT Press, Cambridge (2003)
2. Lindenmayer, A.: Mathematical models for cellular interaction in development: Part I and II. Journal of Theorical Biology 18, 280-299, 300–315 (1968)
3. Turing, A.: The chemical basis of morphogenesis. Philosofical Transactions of the Royal Society B 237, 37–72 (1952)
4. Kitano, H.: Designing neural networks using genetic algorithm with dynamic graph generation system. Complex Systems 4, 461–476 (1990)
5. Hornby, G.S., Pollack, J.B.: Creating high-level components with a generative representation for body-brain generation. Artificial Life 8(3) (2002)
6. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22, 437–467 (1969)
7. Steiner, T., Jin, Y., Sendhoff, B.: A Cellular Model for the Evolutionary development of Lightweight Material with an Inner Structure. In: Proceedings of Genetic and Evolutionary Computation Conferences 2008, Atlanta, pp. 851–858 (2008)
8. Kumar, S.: Investigating Computational Models of Development for the Construction of Shape and Form. PhD Thesis. Department of Computer Science, University College London (2004)

9. Fernandez-Blanco, E., Dorado, J., Rabuñal, J.R., Pedreira, N., Gestal, M.: A Computational Approach to Simple Structure Development. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 825–834. Springer, Heidelberg (2007)
10. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor (1975)
11. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
12. Kephart, J.O., Chess, D.M.: The vision of Autonomic Computing. IEEE Computer Society Press, Los Alamitos (2003), 0018-9162. 41-50

Digital Organ Cooperation: Toward the Assembly of a Self-feeding Organism

Sylvain Cussat-Blanc, Hervé Luga, and Yves Duthen

University of Toulouse - IRIT - UMR 5505
2 rue du Doyen-Gabriel-Marty 31042 Toulouse Cedex 9, France
{Sylvain.Cussat-Blanc,Herve.Luga,Yves.Duthen}@irit.fr

Abstract. In Nature, the intrinsic cooperation between organism's parts is capital. Most living systems are composed of organs, functional units specialized for specific actions. In our last research, we developed an evolutionary model able to generate artificial organs. This paper deals with the assemby of organs. We show, through experimentation, the development of an artificial organism composed of four digital organs able to produce a self-feeding organism. This kind of structure has applications in the mophogenetic-engineering of future nano and bio robots.

1 Introduction

Most living systems are composed of different organs. Cooperation between organs allows them to optimize the exploitation of environmental resources. Its role is crucial for survival in a complex environment. Several works on digital organs development already exist mainly based on two methods: shape generation, which is the most widely discussed, and function generation.

Our previous research dealt with making isolated digital organs. We developed a bio-inspired model able to produce organisms starting from a single cell. The aim was to make an organ library. We now present the cooperation capacity of two kinds of organs: producer-consumers and transfer systems. Assembling these organs produces a self-feeding structure and gives the organism a potentially limitless survival capacity.

The paper is organised as follows. Section 2 gives the related work about artificial development and artificial creature production. Section 3 summarizes the model, already presented in [3]. Section 4 details the experimentation of a self-feeding structure with particular emphasis on environmental parameters. Finally, we conclude by outlining possible future work on this creature.

2 Related Works

Over the past few years, more and more models concerning artificial devolpment have been produced. A common method for developing digital organisms is to use artificial regulatory networks. Banzhaf [1] was one of the first to design such a model. In his work, the beginning of each gene, before the coding itself, is marked by a starting pattern, named "promoter". This promoter is composed of enhancer and inhibitor sites that allow the regulation of gene activations and inhibitions. Another different approach

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 53–58, 2011.

is based on Random Boolean Networks (RBN) first presented by Kauffman [9] and reused by Dellaert [6]. An RBN is a network where each node has a boolean state: activate or inactivate. The nodes are interconnected by boolean functions, represented by edges in the net. Cell function is determined during genome interpretation.

Several models dealing with shape generation have recently been designed such as [5,2,8]. Many of them use gene regulation and morphogens to drive the development. A few produce their own morphogens whereas others use environment "built-in" morphogens. Different shapes are produced, with or without cell specialisation. The well-known French flag problem was solved by Chavoya [2]. This problem shows model specialisation capacity during the multiple colour shifts.

In their models, produced organisms have only one function: filling up a shape. Other models, most often based on cellular automata or artificial morphogenesis (creatures built with blocks), are able to give functions to their organisms [10,7]. Here, creatures can walk, swim, reproduce, count, display... Their goals are either led by user-defined fitness objectives that evaluate the creature responses in comparison to those expected or only led by their capacity to reproduce and to survive in the environment.

The next section presents our developmental model. It is based on gene regulatory networks and an action selection system inspired by classifier rule sets. It has been presented in details in [3].

3 *Cell2Organ*: A Cellular Developmental Model

3.1 The Environment

To reduce simulation computation time, we implement the environment as a 2-D toric grid. This choice allows a significant decrease in the simulation's complexity keeping a sufficient degree of freedom. The environment contains different substrates. They spread within the grid, minimizing the variation of substrate quantities between two neighbouring crosses on the grid. These substrates have different properties such as spreading speed or colour, and can interact with other substrates. Interactions between substrates can be viewed as a great simplification of a chemical reaction: using different substrates, the transformation will create new substrates, emitting or consuming energy. To reduce the complexity, the environment contains a list of available substrate transformations. Only cells can trigger substrate transformations.

3.2 Cells

Cells evolve on the environment's spreading grid. Each cell contains sensors and has different abilities (or actions). An action selection system allows the cell to select the best action to perform at any moment of the simulation. Finally, a representation of a GRN is available inside the cell to allow specialization during division.

Each cell contains different density sensors positioned at each cell corner. Sensors allow the cell to measure the amounts of substrates available on each cell corner. The list of available sensors and their position in the cell is described in the genetic code.

To interact with the environment, cells can perform different actions: perform a substrate transformation, absorb or reject substrates in the environment, divide (see later),

wait, die, etc. This list is not exhaustive. The implementation of the model enables a simple addition of actions. As with sensors, not all actions are available for the cell: the genetic code will give the available action list.

Cells contain an action selection system. This system is inspired by the rule set of classifier systems. It uses data given by sensors to select the best action to perform. Each rule is composed of three parts: (1) The *precondition* describes when the action can be triggered. A list of substrate density intervals describes the neighbourhood in which action must be triggered. (2) The *action* gives the action that must be performed if the corresponding precondition is respected. (3) The *priority* allows the selection of only one action if more than one can be performed. The higher the coefficient, the more probable is the selection of the rule.

Division is a particular action performable if the next three conditions are respected. First, the cell must have at least one free neighbour cross to create the new cell. Secondly, the cell must have enough vital energy to perform the division. The vital energy level need is defined during the specification of the environment. Finally, during the environment modelling, a condition list can be added.

The new cell created after division is completely independent and interacts with the environment. During division, the cell can optimize a group of actions. In nature, this specialization seams to be mainly carried out by the GRN. In our model, we imagine a mechanism that plays the role of an artificial GRN. Each action has an efficiency coefficient that corresponds to the action optimization level: the higher the coefficient, the lower the vital energy cost. Moreover, if the coefficient is null, the action is not yet available for the cell. Finally, the sum of efficiency coefficients must remain constant during the simulation. In other words, if an action is optimized increasing its efficiency coefficient during division, another (or a group) efficiency coefficient has to be decreased.

The cell is specialized by varying the efficiency coefficients during division. A network represents the transfer rule. In this network, nodes represent cell actions with their efficiency coefficients and weighted edges representing efficiency coefficient quantities that will be transferred during the division.

3.3 Obtained Creatures

To find the creature best adapted to a specific problem, we use a genetic algorithm. The creature is tested in its environment. It returns the score at the end of the simulation. Each creature is coded with a genome composed of three different chromosomes: (1) the list of available actions, (2) an encoding of the action selection system and (3) an encoding of the gene regulation network.

Different kinds of creatures have been generated thanks to this model. We can cite for example organs like a transfer system able to transport a substrate from a point to another or morphology developpement like a starfish or jellyfish. Their creatures have been presented in [3]. All creatures have a common property: they are able to repair themselves in case of injury [4]. This feature is an inherent property of the model. It shows the phenotype plasticity of creatures produced.

In the next section, we present the features obtained by producing new organisms and putting them in the same environment. We design an environment wherein the organism will be composed of four organs. Once assembled, their organs will make a

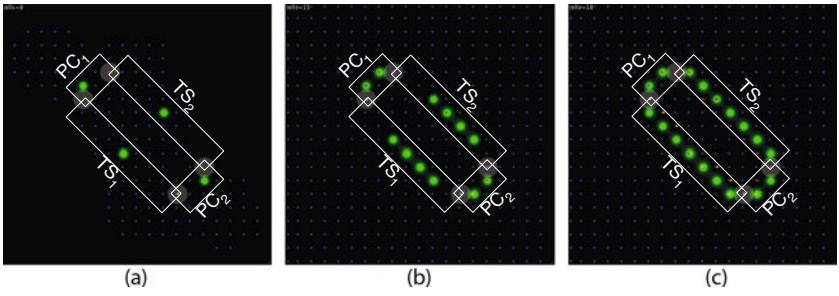


Fig. 1. Result of the cooperation of the 4 different organs. (a) Beginning of the simulation: 4 cells that contain the genetic code of each organ are positioned in the environment. (b) Organ growth: while the 2 producer-consumer organs have finished their development and start their work, the transfer systems continue their growth. (c) All organs have finished their development and a self-feeding structure is made. While producer-consumer organs continue their work, transfer systems start the transfer to feed other organs with new substrates.

self-feeding structure that will allow the organism to maintain its life endlessly. Before that, the organism must develop a sufficient metabolism to start the chain.

4 Experiment: Self-feeding Structure

In order to produce a cycle, the organism is composed of two kinds of organs: transfer systems close to the one presented in [3] and organs able to transform a substrate into another and to position precisely the produced substrate (that will be transferred).

4.1 Description of the Environment

In this experiment, the environment is composed of 3 different substrates: *Water* (represented in blue on the next figures) that will be used by the organism to develop its metabolism, *A and B* (respectively represented in red and yellow on the next figures) substrates that will be used by the organism to produce the self-feeding structure.

Three substrate transformations are available: $Water \rightarrow energy$ produces energy using water ; $A \rightarrow B + energy$ produces B substrate using one unit of A ; $B \rightarrow A + energy$ produces A substrate using one unit of B.

50 units of A substrates are positioned near PC_1 and 50 units of B substrates near PC_2. Organ PC_1 has to transform the substrate A into B and must position it at the entrance of organ TS_1, which transfers the B substrate on the entrance of organ PC_2. Organ PC_2 has to permorm the opposite operation to that of organ PC_1: it transforms B substrate into A and has to put the result at the entrance of organ TS_2, which drives the A substrate back up near organ PC_1. Because all their actions provide energy to the cells, the obtained organism can work endlessly. With the purpose of producing the self-feeding structure, each organ has first been developed individually. The organs have the following list of actions: *divide* (all directions are available), *absorb* and *reject* all kind of substrates and perform one of the *substrate transformations* presented before.

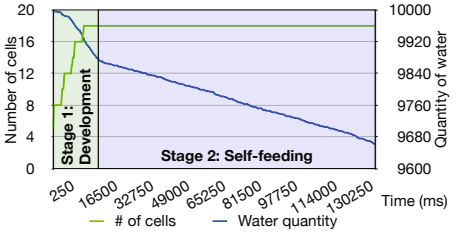


Fig. 2. Number of cells (left ordinate) and quantity of water (right ordinate) in the environment in function of time (abscissa).

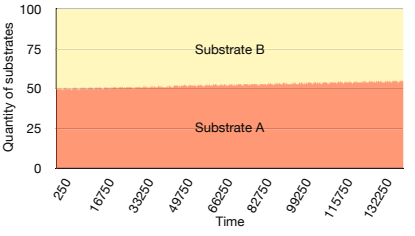


Fig. 3. Evolution of A and B substrate quantity ratio. The B substrate quantity decreases: Organ PC_2 is more efficient than PC_1.

4.2 Results

We compute each different organ in separate environments. Four cells containing the genetic code of their corresponding organ are then added to the environment. They evolve with the aim of generating a self-feeding structure. Figure 1 shows the development and the behaviour of the organism. It is interesting to notice that for each kind of organ, different strategies emerge to reach the goal. For example, organ PC_1 transfers the initial substrate near the goal before transforming it into the final substrate whereas organ PC_2 transforms the substrate before transferring the result to the right place.

The organism obtained is the expected one[1]. The regulation network regulates correctly the size of the transfer systems whereas the organs that transform the substrate develop the different action selection strategies to reach their goals. Organ TS_1 and Organ TS_2 use serial absorption and rejection to move the substrate from the exit of an organ to the entrance of the opposite organ.

Curves presented in figure 2 show the evolution of the number of cells and the water quantity in the environment. Water quantity strongly decreases at the beginning of the simulation, before the initialisation of the cycle (stage 1). Different organs use water to start their metabolism. When the cycle starts (stage 2), organs use the cycle as metabolism. Organs still consume water to produce energy that will be stocked for the future. The curve presented in figure 3 shows the ratio of A and B substrates. B substrate quantity slowly decreases. This proves an efficiency difference in the organs: organ PC_1 converts A into B more slowly than organ PC_2 does the opposite. Even if the difference is small, this curve shows that the cycle is not endless: after a long period of time, B substrate will disappear and the cycle will be broken.

5 Conclusion and Future Works

In this paper, we present an original result of the developmental bio-inspired model *Cell2Organ*. After making an artificial organ library, we test cooperation between organs. The experimentation shows the development of an artificial organism, composed

[1] Videos of this organism development and of each organ functioning seperately are available on the website http://www.irit.fr/∼Sylvain.Cussat-Blanc

of four digital organs. The cooperation of its organs creates a self-feeding structure. This kind of structure, with the self-repairing properties presented in [4], could be interesting for a morphogenetic-engineering approach of future bio and nano robots.

Continuations of this work are multiple. First of all, we are currently starting the development of the organism with four cells, one for each organ. We want to develop the organism starting from only one cell. With this purpose, we are working on a "pre-organism" able to position cells on the four starting positions of the final organs. The organism will have to switch it genome to the different organs' genomes and, finally, to resorb itself so as not to interfere with the organism's evolution.

We are also working on making different layers of the simulated environment. A physical layer will allow us to develop our organism at the same time in a physical world, with all its properties and the current "chemical" world to maintain the metabolism of the creatures. A hydrodynamic layer will simulate substrate diffusions more efficiently. For example, this layer will allow a cell to expulse a substrate with a chosen strength to position it in a particular place. It will also simulate fluid flows. Cells will have to adjust their behaviour according to new data.

Acknowledgment. Experiments presented in this paper were carried out using ProActive, a middleware for parallel, distributed and multi-threaded computing (see http://proactive.inria.fr), and the Grid'5000 French experimental testbed (see https://www.grid5000.fr).

References

1. Banzhaf, W.: Artificial regulatory networks and genetic programming. Genetic Programming Theory and Practice, 43–62 (2003)
2. Chavoya, A., Duthen, Y.: A cell pattern generation model based on an extended artificial regulatory network. Biosystems (2008)
3. Cussat-Blanc, S., Luga, H., Duthen, Y.: From single cell to simple creature morphology and metabolism. In: Artificial Life XI, pp. 134–141. MIT Press, Cambridge (2008)
4. Cussat-Blanc, S., Luga, H., Duthen, Y.: Cell2organ: Self-repairing artificial creatures thanks to a healthy metabolism. In: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE CEC 2009 (2009)
5. de Garis, H.: Artificial embryology and cellular differentiation. In: Bentley, e.P.J. (ed.) Evolutionary Design by Computers, pp. 281–295 (1999)
6. Dellaert, F., Beer, R.: Toward an evolvable model of development for autonomous agent synthesis. In: Artificial Life IV. MIT Press, Cambridge (1994)
7. Garcia Carbajal, S., Moran, M.B., Martinez, F.G.: Evolgl: Life in a pond. In: Artificial Life XI, pp. 75–80 (2004)
8. Joachimczak, M., Wróbel, B.: Evo-devo in silico: a model of a gene network regulating multicellular development in 3d space with artificial physics. In: Artificial Life XI, pp. 297–304. MIT Press, Cambridge (2008)
9. Kauffman, S.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theorical Biology 22, 437–467 (1969)
10. Sims, K.: Evolving 3d morphology and behavior by competition. In: Artificial Life IV, pp. 28–39 (1994)

Distance Discrimination of Weakly Electric Fish with a Sweep of Tail Bending Movements

Miyoung Sim and DaeEun Kim

Biological Cybernetics Lab
School of Electrical and Electronic Engineering,
Younsei University,
Schinchon, Seoul, 120-749,
Corea, South Korea
{simmi,daeeun}@yonsei.ac.kr

Abstract. Weakly electric fish use active electrolocation to identify objects. They generate electric field by the electric organ discharge and perceive the distortion of electric image with existence of certain object. There have been many researches to comprehend the electrolocation mechanism of electric fishes. It is known that the ratio between the maximal slope of electric image and its maximal amplitude can discriminate object distances, regardless of object size and conductivity. In this paper, we suggest that the temporal pattern with tail bending is another cue to disciminate object distances. As a result, the electric field pattern for a specific electroreceptor shows consistency, regardless of object size and conductivity, when the distance is constant. Also, the lateral location of an object significantly changes the temporal pattern of electric image.

Keywords: electric fish, electrosensory system, electrolocation, biorobotics.

1 Introduction

Weakly electric fish generate electric field by the electric organ and identify objects with electroreceptors that sense the change of electric field. The mechanism of electrolocation is related with interpretation of electric image. The electric image can be regarded as a stimulus image that represent the perceived stimulus through sensory cells [1].

One of challenging issues in the electrolocation is how electric fish disciminate object distances. Rasnow [2] studied the distortion of electric image when spherical objects are in the electric field generated by weakly electric fish, *Apteronotus leptorhynchus*. The simulation result of electric image along the skin showed there is certain relationship between electric images and object parameters, such as rostrocaudal position, lateral distance, and conductance. Chen et al. [3] argued that a rostrocaudal position of a given object along the fish body fixes the location of peak amplitude of electric image, and another features of the object such as size and distance influence the maximum level of amplitude.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 59–66, 2011.

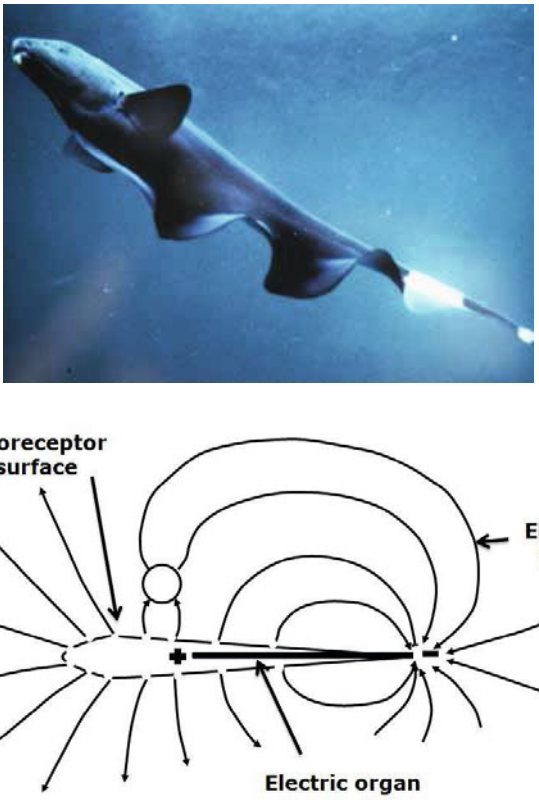


Fig. 1. Diagram for electrolocation of electric fish (black bar indicates the electric organ) (Source of the photograph is from www.fieldmuseum.org/research_collections/ecp/ecp_sites/parker_gentry/blghost.htm and the diagram is modified from [8])

Some researchers [4,5] suggested there exists a cue for distance discrimination. The ratio between maximal image slope and maximal image amplitude can measure the distance of most objects, independently of size, material and shape. Sicardi et al. [6,7] also argued that this relative slope is independent of the object size and conductivity.

So far many researchers have handled object localization of electric fish with a spatial distribution of electroreceptors. The experiments suggest that the relative activation of electroreceptors with a target object can achieve the distance discrimination [5]. In this paper, we investigate temporal information of electrosenses with a sweep of tail bending movement for the distance discrimination. We argue that tail bending triggers the same temporal pattern of electric image at a given electroreceptor with a fixed distance of object, independently of size and material.

2 Method

2.1 Modeling Electric Field

Our simulation model follows Chen et al.'s model [3] and Rasnow's model [2] to estimate the electric field intensity and transdermal potential differences on the skin of electric fish. We assumed that the fish body length is set to 21cm and positive poles are uniformy distributed with 10 poles per 1cm. One negative pole and a number of positive poles form the electric field. Electric potential at the location x, caused by these electric poles can be calculated as follows:

$$V(\boldsymbol{x}) = \sum_{i=1}^{m} \frac{q/m}{|\boldsymbol{x} - \boldsymbol{x}_p^i|} - \frac{q}{|\boldsymbol{x} - \boldsymbol{x}_p^i|} \tag{1}$$

where m is the number of positive poles and x_p^i is the location of the i-th pole, and q is the normalized potential value. From this, the electric field has a weakly positive field over most of the side skin and relatively strong negative field at the end of tail. The electric field can be derived from the gradient of electric potential.

$$E(\boldsymbol{x}) = -\nabla V(\boldsymbol{x}) = \sum_{i=1}^{m} \frac{q/m}{|\boldsymbol{x} - \boldsymbol{x}_p^i|^3}(\boldsymbol{x} - \boldsymbol{x}_p^i) - \frac{q}{|\boldsymbol{x} - \boldsymbol{x}_p^i|^3}(\boldsymbol{x} - \boldsymbol{x}_p^i) \tag{2}$$

The potential perturbation of spherical objects [2] can be given as follows:

$$\delta V(\boldsymbol{x}) = \chi \frac{a^3 E(\boldsymbol{x}_{obj}) \cdot (\boldsymbol{x} - \boldsymbol{x}_{obj})}{|\boldsymbol{x} - \boldsymbol{x}_{obj}|^3} \tag{3}$$

where a is the radius of object and χ is the electrical contrast. The electric contrast is related to the material, which is -0.5 for a perfect insulator and one for a perfect conductor. If the impedance of the object exactly matches with water, then the electrical contrast is zero.

The transdermal potential difference $V_{td}(x_s)$ is related with the normal component on the skin surface.

$$V_{td}(\boldsymbol{x}_s) = E(\boldsymbol{x}_s) \cdot \hat{n}(\boldsymbol{x}_s) \frac{\rho_{skin}}{\rho_{water}} \tag{4}$$

where x_s is the position of sensor and $\hat{n}(x_s)$ is the normal component for an electroreceptor.

Electroreceptors are distributed on the skin surface of electric fish. Here, we assume electroreceptors are uniformly distributed and the normal vector at the skin is orthogonal to the body axis, to simplify the analysis of electric image. We examine the temporal change of the transdermal potential difference when the tail bends. The tail bends from $-45°$ to $45°$ approximately [3]. Bending tail draws a circular arc around the pivot. A curvature radius R is $L\theta$, where L is the tail length and θ is the bending angle. In this paper we investigate the temporal change at each sensor with a variety of object sizes, conductivities, lateral distances, and rostrocaudal distances.

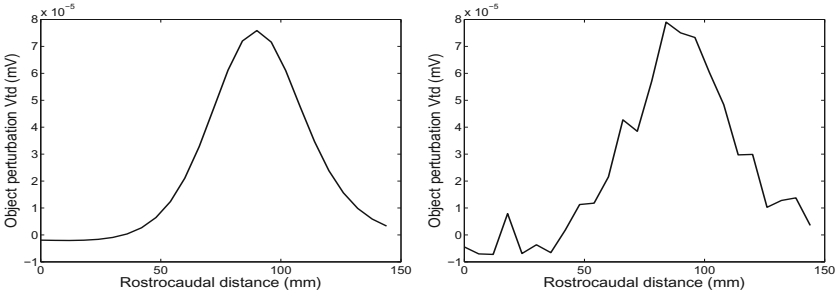


Fig. 2. Potential perturbation curve (a) figure without noise term (b) figure with random noise

2.2 Relative slope

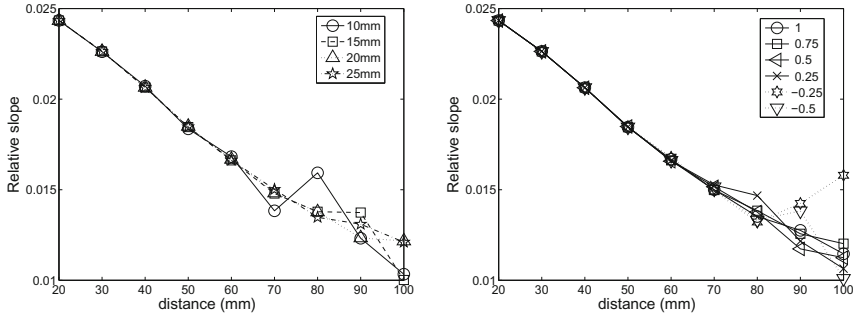
We can obtain the electric images to reflect the transdermal potential values on the skin surface. The ratio of transdermal potential values without any object, to the perturbation of electric field with an object composes the modulation as electric image measurements [4,5]. In our simulation, uniform random noise is added to the electric image. This noisy image has been filtered through the butterworth low pass filter. The ratio between maximal slope and maximal amplitude of the electric image, called *relative slope*, can be exploited to determine the distance of an object.

3 Experiments

3.1 Relative Slope

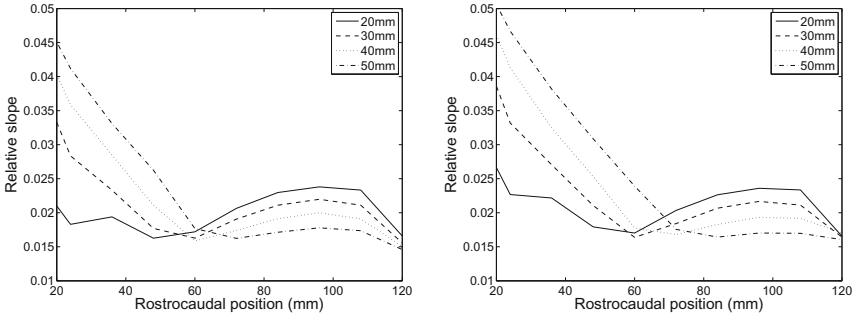
We first tested the relative slope property of electric image. Fig. 3 shows the simulation result for the relative slope over varying distances with sphere objects. From the figures, we can easily observe that the relative slope is a good measure to discriminate the distance, irrespective of object size and conductivity. This agrees with von der Emde et al.'s result [4]. The relative slope for a small size of objects or long distances are greatly influenced by noise, since their signal levels as well as the maximum amplitudes are relatively small. The relative slopes at long distances show more deviations from the normal slope pattern.

Through the simulation, it is apparent that the relative slope indicates the object distance. However, the relative slope is affected by the rostrocaudal position of a target object as well as tail bending angle. Fig. 4 shows the relative slope changes depending on the rostrocaudal position and tail bending movement. Thus, we need prior information of the rostrocaudal position and bending angle to determine the distance of an object.



(a) (b)

Fig. 3. Relative slope with object size and conductivity (a) relative slope with varying object sizes and constant conductivity (0.5) (b) relative slope with varying conductivities and constant object size (sphere object with a radius of 2cm)



(a) (b)

Fig. 4. Relative slope with varying object positions (rostrocaudal positions); a sphere object with a radius of 2cm and conductivity 0.5 is tested at the lateral distance of 20 mm, 30 mm, 40 mm and 50 mm (a) relative slope when the tail is not bent (b) relative slope with the tail-bending angle 45°

3.2 Temporal Pattern

We investigate another cue to discriminate the object distances, which is based on temporal pattern of electric image. The relative slope mentioned above evaluates the spatial distribution of electrosenses along a set of electroreceptors on the skin surface. In contast, the temporal pattern at a given electroreceptor with a sequence of tail bending movements provides another viewpoint to estimate the distance.

We simulated the electric image with a sweep of tail movements, where the tail bends from −45° to 45° and again from 45◦ to −45°. Temporal patterns of potential perturbation with sphere objects can be obtained from the negative pole shift. The signals are normalized into the scale [0, 1], because the signal

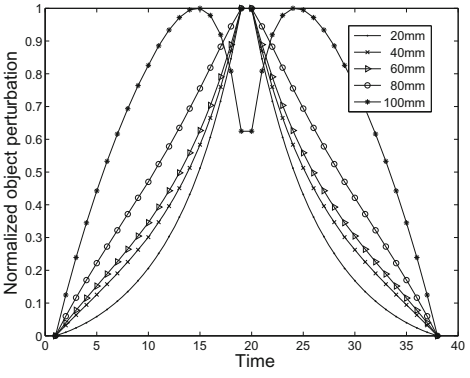


Fig. 5. Temporal pattern depending on lateral distances; a sphere object with radius 2cm was tested

amplitude varies depending on sizes and distances. In our simulation experiments, the object size varies from 0.5cm to 2.5cm with interval 0.5cm, and the conductivity changes from 1 to -0.5 with interval 0.25. Also, the object was shifted along the rostrocaudal direction and the corresponding electric image was calculated. We measured the sensor activations of three different electroreceptors at 4cm, 12cm, 20cm from the mouth, respectively.

Fig. 5 shows the lateral distance significantly changes the temporal pattern. We checked if another factors are involved with the pattern. Interestingly, Fig. 6 shows that the conductivity of objects have no effect on the temporal pattern for a sweep of tail movement. The negative conductivity only changes the orientation of temporal pattern, which leads to reverse orientation. The shape itself does not change regardless of the conductivity level. The two patterns for positive and negative conductivity have a mirror image each other. In addition, each electroreceptor has the same temporal pattern for varying sizes of objects as shown in Fig. 7.

In contrast, Fig. 8 displays that the rostocaudal position of a target object significantly changes the temporal pattern. From the above experiments, we can infer that the distances of a target object is directly involved with the temporal pattern by a sweep of tail movement, irrespective of conductivity and size of the object. The maximal slope in the normalized temporal pattern can be a cue to discriminate object distances. The temporal pattern can be easily tested with a few electroreceptors, while estimating the relative slope requires a large distribution of electroreceptors on the skin surface.

The relative slope and the temporal variation gives us the information about the position of a target object. This measure is independent of object size and conductivity, but affected by rostrocaudal position and lateral distance from the midline of electric fish. However, the rostrocaudal postion of a target object is identified easily from the peak amplitude position in the electric image. If the rostrocaudal position of a target object is determined in advance, the temporal

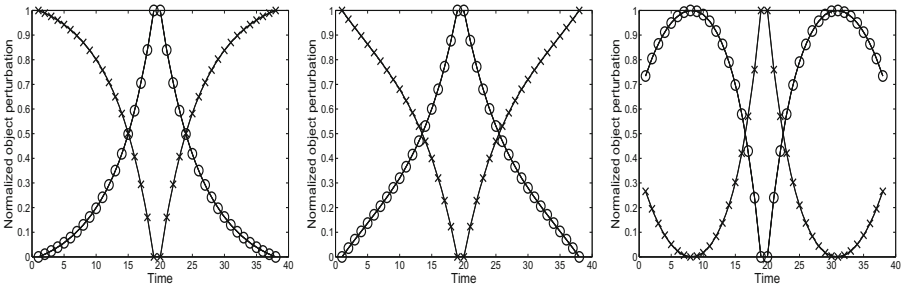


Fig. 6. Temporal patterns with varying conductivities (from the electrosensor values at a distance of 4cm, 12cm, 20cm from the mouth, respectively); sphere objects with a radius of 2 cm were tested at the fixed distance of 5 cm ('o': positive conductivity and 'x': negative conductivity)

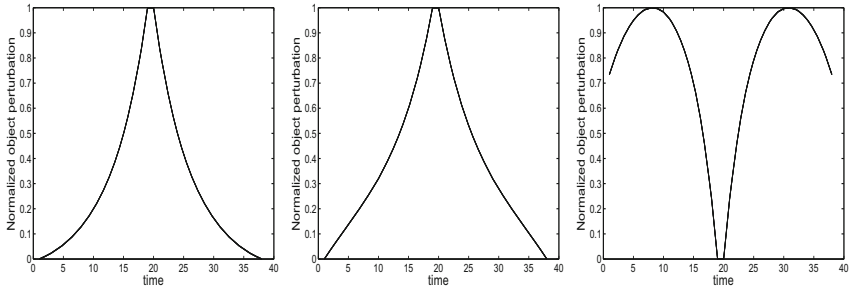


Fig. 7. Temporal patterns with varying sizes (from the electrosensor values at a distance of 4cm, 12cm, 20cm from the mouth, respectively); sphere objects with the conductivity 0.5 were tested at the fixed distance of 5 cm (sphere objects with varying radius sizes from 0.5cm to 2.5cm)

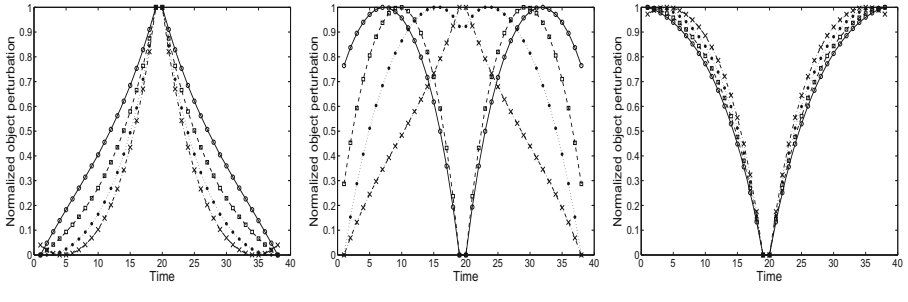


Fig. 8. Temporal patterns with varying rostrocaudal positions of an object (from the electrosensor values at a distance of 4cm, 12cm, 20cm from the mouth, respectively); the sphere object was tested at varying rostrocaudal distances of 2.4cm (solid), 3.6cm (dashed), 4.8cm (dotted), 6.0cm (dotdashed) from the mouth and at the fixed lateral distance of 5cm

pattern or the relative slope can be used to discriminate the lateral distance of the object. Here, we do not argue which measure between the relative slope and the temporal pattern is more effective to discriminate the object distance. It is presumed that both measures depending on environmental situation help the electric fish to guess the distance of a target object.

4 Conclusion

The relative slope, the ratio between the maximal slope and the maximal amplitude of electric image over a distribution of electroreceptors, is a useful parameter to discriminate the object distances, independently of size and conductivity of the object. For object localization, the location of peak amplitude is the direction of an object and the relative slope provides the lateral distance as biologists argued. In this paper, we suggest the temporal pattern even at one electroreceptor also provides a cue to determine the object distance, irrespective of size and conductivity. Active sensing allows to extract another type of localization information. Possibly the object localization can be extended to shape recognition of objects with electrolocation. For the future work, this kind of active sensing approach with a few electrosensors can be applied to s mobile robots to search for underwater objects.

Acknowledgments. This work was supported by the Korea Science and Engineering Foundation(KOSEF) grant funded by the Korea government(MEST) (No.2009-0080661).

References

1. Caputi, A., Budelli, R.: Peripheral electrosensory imaging by weakly electric fish. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 192(6), 587–600 (2006)
2. Rasnow, B.: The effects of simple objects on the electric field of Apteronotus. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 178(3), 397–411 (1996)
3. Chen, L., House, J., Krahe, R., Nelson, M.: Modeling signal and background components of electrosensory scenes. Journal of Comparative Physiology A: Sensory, Neural, and Behavioral Physiology 191(4), 331–345 (2005)
4. Von der Emde, G., Schwarz, S., Gomez, L., Budelli, R., Grant, K.: Electric fish measure distance in the dark. Nature 395(6705), 890–894 (1998)
5. Von der Emde, G.: Active electrolocation of objects in weakly electric fish (1999)
6. Sicardi, E., Caputi, A., Budelli, R.: Physical basis of distance discrimination in weakly electric fish. Physica A: Statistical Mechanics and its Applications 283(1-2), 86–93 (2000)
7. Schwarz, S., von der Emde, G.: Distance discrimination during active electrolocation in the weakly electric fish Gnathonemus petersii. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 186(12), 1185–1197 (2001)
8. Heiligenberg, W.: Neural nets in electric fish. MIT Press, Cambridge (1991)

Adaptation in Tissue Sustained by Hormonal Loops

Dragana Laketic and Gunnar Tufte

Norwegian University of Science and Technology, Trondheim 7491, Norway

Abstract. When faced with the requirements for the design of an autonomous adaptive system, many aspects of the system organisation need be addressed. In living systems, the co–evolution with the environment has provided the solution for such challenges in a form of inherent mechanisms which are employed when the environmental fluctuation occurs leading to the organism achieving adaptation through some adaptive process. In this paper we investigate such mechanisms, more precisely the priniples on which their operation is based. In particular, the focus is set on endocrine system within homeostatic processes. We postulate that adaptation to a fluctuating environment can be achieved if initiated and sustained by the hormone flow loops. Such statement is further supported by simulations. Based on the recognised advantages of the system organisation endowed with the ability to form hormonal loops, the avenues of research are identified for further work.

1 Introduction

In the beginning there were catalytic cycles out of which the life emerged and ever since, the cyclic processes have not stopped to weave more and more complex creations. Cyclic processes can be recognised at different levels of the living systems' organisation and it can be said that they constitute the very basis of life. The life–related processes presume interaction between the living system and its environment [1,2] and it has been widely accepted that living systems have not evolved but rather co-evolved with their environment [3,4]. As a result, the living systems are capable of surviving within the environment despite its fluctuating nature. When a fluctuation occurs, inherent mechanisms are employed and the organism performs some structural or functional change – it adapts to environmental fluctuation. An example of these mechanisms is represented by a whole plethora of homeostatic processes. The body's environment is continuously monitored – sensed and, accordingly, adaptive processes are initiated, sustained and regulated resulting in the preservation of the body's internal equilibrium – homeostasis. In doing so, various systems available in the body are engaged [5] which are interrelated and interwoven in their operation. A prominent place belongs to the endocrine system for its communication and control role. It exhibits ability to control processes within the living system, yet with the employment of just tiny amounts of substances – hormones, for this purpose.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 67–74, 2011.

Adaptive processes within living systems are addressed in a somewhat simplified view through the formalism of cybernetics [6]. In this view, the dynamics of adaptive processes is recognised as regulated by feedback loops. In our investigation, we postulate that the adaptation process can be initiated and sustained by the formation of feedback loops of hormone flows within the system. Such statement is further supported with simulations on a model of a modular system. The results not only show successful adaptation, they also open new research avenues towards the enhancement of the models of processes within some tissue so as to perform adaptation task. Furthermore, considerations for the implementation of such adaptive system within some modular man–made system are also presented.

2 Learning and Adopting from Living Systems

Ever-present environmental fluctuations influence physiological processes within the living organism. Such processes are vital for the organism and therefore its viability comes from the ability to adapt to a fluctuating environment. As mentioned, endocrine system, a system of organs and tissues which secrete special substances – hormones, is prominent for the control and communication roles within homeostatic processes. It reacts to the sensed fluctuation in the environment by the secretion of special messengers – hormones, which are carried around the body via bloodstream thereby reaching all the cells in the living system. However, only the cells possessing the matching *receptor* will react to the hormone. Tiny amounts of hormones can produce avalanche of reactions which contribute to the system's preservation of homeostasis. However tiny the amounts of secreted hormones may be, they are regulated through special mechanisms based on feedback loops. Feedback loops are closed between the place of the hormone secretion i.e. an endocrine gland or some tissue, and the target tissue which the hormone affects [5]. Once secreted, hormones do not remain in the organism forever. They have their lifetime after which they are cleared from the organism through various mechanisms.

It can be said that the adaptation process in living systems occurs in stages, the first being creation of inherent adaptive mechanisms as a result of co-evolution with environment and the second employment of these mechanisms when the change occurs, as has been recognised and studied from a more mechanistical standpoint in [7]. There, the investigated system preserves its homeostasis by keeping *essential variables* within certain limits despite the disturbance coming from the environment. Homeostasis for adaptation has been studied by many researchers. Ashby has demonstrated homeostatic adaptation in practice in the *homeostat*, a modular electro–mechanical structure. Gaian theory [3], views the whole biosphere as a homeostatic system. Similar ideas may be found in [8,9,10]. Within homeostatic processes, endocrine system has been investigated in evolutionary robotics [11,12,13], modular self–reconfigurable robots [14], multiprocessor system for fault tolerance [15].

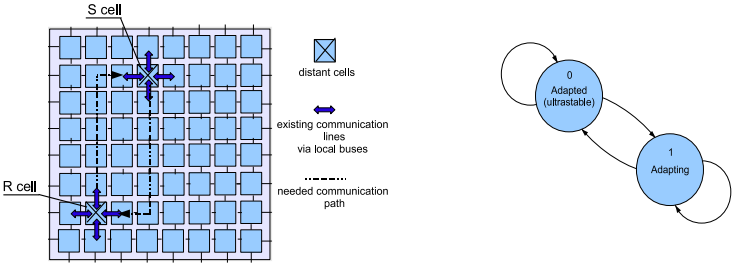
3 The Model of the System Under Investigation

In order to simulate adaptive processes, a model of an adaptive man–made system was created. It consists of a number of cells placed in a grid formation as in figure 1(a) each of which is connected to its four immediate neighbours (east, south, west and north) via local connections. These connections transmit not only computational data but also the signals further referred to as hormones, the name chosen due to the role they perform in the system – the messengers. The assumed architecture lends itself to analysis through the formalism of cellular automata, CA [16,17].

For each cell it is assumed that it performs some functionality which can be adjusted by the set of tuning parameters. Further, it possesses a sensor for sensing the fluctuations in its local environment i.e. variations in some environmental parameter – temperature, pressure, to give a few less abstract examples.

The system is configured so as to perform certain functionality. Configuration is implemented in a similar way as in [18] where each cell possesses two types of identifiers – *physical ID* and *encoding ID*. The first refers to the cell's position within the grid and the second to the cell's functional connectivity. In relation to its *encoding ID*, i, the cell receives inputs from the cell with the *encoding ID* $i-1$ and yields output to the cell with the *encoding ID* $i+1$. As an example, figure 3(a) shows *encoding ID* values for one possible configuration.

Taking somewhat mechanistic approach, the system can be said to be in one of the two states: it is either adapted (state 0) or adapting (state 1) to its environment, as shown in figure 1(b). When in state 0, each cell performs functionality according to the cell's tuning parameters which correspond to its local environment thereby contributing to the system performing correct functionality. On the other hand, during adaptation process, tuning parameters within one or more of the system's cells do not correspond to its local environment and therefore the functionality of the system as a whole deviates from the desired. In this state, such cells go through the adaptation process in order to tune these parameters.



(a) Schematic view of the investigated architecture with the hormonal loop between S and R cell

(b) System States

Fig. 1. Schematic of the assumed system architecture and system state diagram

In [19] we have shown how hormonal loops can initiate and sustain adaptation in case of stochastic environmental fluctuations. Here, we briefly present the formation of hormonal loops. Within the assumed system, the hormonal loops are made out of two types of hormones: *S hormone*, which is secreted by the cell whose local environment has changed (*S cell*) and *R hormone*, secreted by the *S cell*'s functionally related cell (*R cell*). We have chosen to refer to the cells which secrete hormones as *S cell* and *R cell* for their 'Sensing the environmental fluctuation' and 'Recognising incoming *S hormone*' roles respectively. The following sequence of events leads to the closing of the hormone flow loop:

- the cell (*S cell*) senses the environmental fluctuation
- *S cell* begins secreting *S hormone*
- *S hormone* reaches functionally related cell (*R cell*)
- *R cell* recognises *S hormone*
- *R cell* begins secreting *R hormone*
- *R hormone* reaches *S cell*

The amount of hormone is assumed to vary during its lifetime according to the exponential decay function and the hormone is considered to be present in the system as long as its amount is larger than some minimum value, *min*:

$$Q(n) = e^{-n/\tau} \qquad Q(n) > min \tag{1}$$

The presence of both these hormones is a necessary condition for the adaptation process to occur in the *S cell*. Adaptation process is performed through the adjustments of the cell's tuning parameters until they reach the values corresponding to its local environment leading to the cell performing correct functionality. As shown in [19], the hormonal loops can be sustained for the needed amount of time if the decay rate parameter τ in equation 1 is adjusted according to the estimate of functional deviation in the functionally related cell.

Figure 2 shows the state diagram which describes the cell's behaviour. It is an extension of the state diagram presented in [19] to account for the flow of multiple hormones. The following control variables are monitored to determine the cell's state transitions: the change in the environmental parameter (EE), the cell's *S hormone* present (SM), the cell's *R hormone* present (RM), hormones from functionally unrelated cells present (HO), the incoming S hormone recognised (RR), the incoming R hormone recognised (FB). The hormone is recognised by the cell if it comes from the functionally related cell. Control variables are omitted from the figure for the sake of clarity.

The cell's state depends on the presence or absence of hormone(s) (H), the functionality it performs (A) and the value of the environmental parameter (E) which is measured locally and whose variations represent environmental fluctuations. Therefore, the cell's state is represented as a 3-tuple (H, A, E) for which possible values are as follows:

– H: L - no hormone present, S - sending *S hormone*, R - sending *R hormone*, P - passing hormone not functionally related to the cell, SP - both S and P, PR - both P and R;

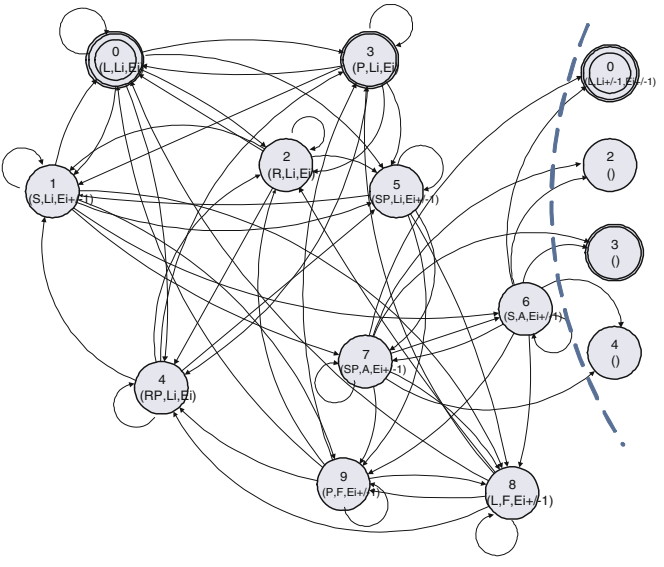


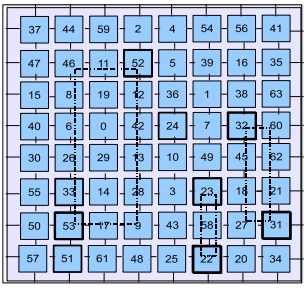
Fig. 2. The cell's state diagram

- A: L0, L1, L2, L3 or L4 - functionality adapted to E0, E1, E2, E3 and E4 respectively, A - adapting or F - failed to adapt;
- E: E0, E1, E2, E3 and E4 - five different values of the environmental parameter under consideration.

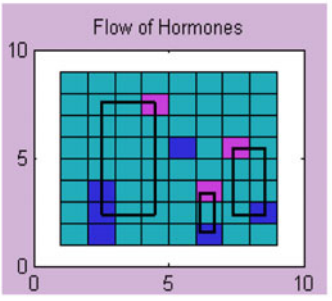
Figure 2 shows the cell's state diagram only for one set of the states corresponding to one change of the environmental parameter i.e. from E_i to E_{i+1}. The broader state diagram includes multiple such regions according to the number of possible discrete values of the environmental parameter E. In comparison to figure 1(b), it can be said that the system is in state 0 when each cell is in one of the states 0 or 3. In these states, the cell does not contribute to any functional deviation of the system as a whole. While one or more cells are in any of the states pertaining to adaptation process, the system is in state 1.

4 Experimental Setup and Simulation Results

The main goal of the presented simulations is to demonstrate how creation of hormonal loops initiated by environmental fluctuation can further initiate and sustain adaptation process within the system tissue until adaptation is achieved. For that purpose, the model of a system introduced in section 3 was used. It has been monitored for certain number of discrete time ticks during which environmental parameter of randomly chosen cells was changed. The focus is on monitoring hormonal loops created between functionally related cells which initiate and sustain adaptation process.



(a) Hormonal loops between functionally related cells for the three S cells with sensed variation in environment (52, 23, 32)



(b) Tick 33: Plot of the hormonal flow within the system (loops are additionally marked)

Fig. 3. Hormonal loops sustaining adaptation

The simulations were run for the configuration of the system as shown in figure 3(a). The system states during simulations were monitored and shown as sets of plots, one for each element of the 3–tuple which represents the cell's state. Different values of the parameter are assigned different colours so that the adapted system has 'Functionality' plot (A) and 'Environmental parameter' plot (E) with the matching colours for each cell. For the sake of clarity in the results presented in figure 3(b), only three cells, shown in magenta when in state H:S, have the value of the environmental parameter changed:

1. tick 1: cell (1,3), *encoding ID* 52, changes parameter E_0 to E_1
2. tick 10: cell (5,5), *encoding ID* 23, changes parameter E_0 to E_1
3. tick 20: cell (3,6), *encoding ID* 32, changes parameter E_0 to E_1
4. tick 25: cell (1,3), *encoding ID* 52, changes parameter E_1 to E_2

In this figure, it can be clearly seen that the loops of hormone flows have been formed between these cells and their functionally related cells, shown in blue when in state H:R. Monitoring of the simulation runs shows that the adaptation process is initiated and sustained by these hormonal loops until adaptation is achieved.

5 Discussion

Seeking inspiration for an effective adaptive technique for autonomous, adaptive man–made systems, we have investigated principles of adaptation initiated and sustained by the loops of hormone flows which may be recognised in living systems. The behaviour of the investigated system is such that it maintains its homeostasis in the presence of environmental fluctuations which can endanger system's functionality. The sensed variation in environmental parameter initiates hormone flows around the architecture which, upon closing the loops, initiate

and sustain adaptation process. Due to such processes, the system's functionality, endangered by environmental fluctuation, is preserved.

It can be argued if such preservation of functionality is an emergent property. Is viability of the tissue endowed with the ability to form hormonal loops emergent in the presence of environmental fluctuation? Functionality of the system is a property of the system as a whole. On the other hand, the hormone flow is determined by the cell's behaviour and each cell has only connections to its immediate neighbours. However, the messaging system inspired by hormonal communication and control makes it possible for functionally related cells to communicate and contribute to the maintenance of functionality irrespective of their placement within the architecture. So, the system's functionality will not be lost despite fluctuating environment – maintenance of its functionality will emerge as a result of the processes supported by hormonal loops.

Hormonal loops are formed between the cells i.e. at the higher hierarchical level of organisation than the cell-level. It is a challenging task to investigate this phenomenon further for the self-sustainability of the multicellular system. In particular, further investigation should address models of self-referring and autonomous tissue – the qualities we sought in adaptive and autonomous man–made system. For such investigation, artificial chemistries (AC) [20] represent suitable tools for further experimenting. AC systems, already recognised as tools for emergence–based technologies, can be endowed with hormonal flows initiated by environmental fluctuations thereby setting the grounds for the development of a novel system for adaptive information processing within fluctuating environment.

Another challenge is deeper investigation into possible technologies which could support implementation of the adaptive system which exhibits presented principles of adaptation. When we look into today's electronics systems, it has already been recognised that silicon technology will soon not be able to meet the needs of the near–future computational demands (the wall will be met soon!). Therefore, new technologies will necessarily take its place [21], [22]. As it appears, all these technologies are extremely sensitive to environmental fluctuations – a small variation in temperature or pressure, for example, may lead to completely undesired outcome of the computation based on some chemical reaction. Endowed with the system which enables its tissue to adapt to such fluctuating environment, they would make a promissing alternative.

References

1. Maturana, H., Varela, F.: Autopoiesis and Cognition: the Realization of the Living. D. Reidel Publishing Co., Dordecht (1973)
2. Ganti, T., Griesemer, J. (contributor), Szathmary, E. (contributor): The Principles of Life. Oxford University Press, Oxford (2003)
3. Lovelock, J.: Gaia - A new look at life on Earth. Oxford University Press, Oxford (1979)
4. Capra, F.: The Web of Life. Anchor Books (division of Random House, Inc.) (1996)
5. Guyton, A., Hall, J.: Textbook of Medical Physiology. Elsevier Saunders (2005)

6. Wiener, N.: Cybernetics or Control and Communication in the Animal and the Machine. The Technology Press/John Wiley and Sons Inc., Cambridge/Chichester (1948)
7. Ashby, W.R.: Design for a Brain, the origin of adaptive behaviour. Chapman & Hall Ltd., Sydney (1960)
8. Dyke, J., Harvey, I.: Hysteresis and the limits of homeostasis:from daisyworld to phototaxis. In: Capcarrère, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., Timmis, J. (eds.) ECAL 2005. LNCS (LNAI), vol. 3630, pp. 241–246. Springer, Heidelberg (2005)
9. Neal, M., Timmis, J.: Timidity: A useful emotional mechanism for robot control? In: Informatica – special issue on perception and emotion based control, The Slovene Society Informatika, Ljubljana, Slovenia, pp. 197–204 (2003)
10. Moioli, R., Vargas, P., Von Zuben, F., Husbands, P.: Towards the evolution of an artificial homeostatic system. In: Proceedings of IEEE Congress on Evolutionary Computation 2008, pp. 4023–4030 (2008)
11. Vargas, P.A., Moioli, R.C., de Castro, L.N., Timmis, J., Neal, M., Von Zuben, F.J.: Artificial homeostatic system: A novel approach. In: Capcarrère, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., Timmis, J. (eds.) ECAL 2005. LNCS (LNAI), vol. 3630, pp. 754–764. Springer, Heidelberg (2005)
12. Neal, M., Feyereisl, J., Rascunà, R., Wang, X.: Don't touch me, i'm fine: Robot autonomy using an artificial innate immune system. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 349–361. Springer, Heidelberg (2006)
13. Di Paolo, E.A.: Organismically-inspired robotics: Homeostatic adaptation and natural teleology beyond the closed sensorimotor loop. In: Advanced Knowledge International on Dynamical systems approach to embodiment and sociality, Adelaide, Australia, pp. 19–42 (2003)
14. Shen, W.M., Salemi, B., Will, P.: Hormone-inspired adaptive communication and distributed control for conro self-reconfigurable robots. IEEE Transactions on Robotics and Automation, 700–712 (2002)
15. Greensted, A.J., Tyrrell, A.M.: Implementation results for a fault-tolerant multicellular architecture inspired by endocrine communication. In: Proceedings of NASA/DoD Conference on Evolvable Hardware (2005)
16. Sipper, M.: Evolution of Parallel Cellular Machines, The Cellular Programming Approach. Springer, Heidelberg (1997)
17. Floreano, D., Mattiussi, C.: Bio–Inspired Artificial Intelligence. The MIT Press, Cambridge (2008)
18. Macias, N., Durbeck, L.: Adaptive methods for growing electronic circuits on an imperfect synthetic matrix. In: Bio Systems, pp. 173–204. Elsevier, Amsterdam (2004)
19. Laketic, D., Tufte, G., Haddow, P.C.: Stochastic adaptation to environmental changes supported by endocrine system principles. In: NASA/ESA Adaptive Hardware and Systems, AHS 2009 (2009) (accepted)
20. Dittrich, P., Ziegler, J., Banzhaf, W.: Artificial chemistries – a review. In: Artificial Life, pp. 225–275. Massachusetts Institute of Technology (2008)
21. Fair, R.B.: Digital microfluidics: is a true lab-on-a-chip possible? In: Microfluidics and Nanofluidics, pp. 245–281. Springer, Heidelberg (2007)
22. Shukla, S.K., Bahar, R.I. (eds.): Nano, Quantum and Molecular Computing. Kluwer Academic Publishers, Dordrecht (2004)

Embodiment of the Game of Life

Kohei Nakajima[1] and Taichi Haruna[2]

[1] Department of General Systems Sciences, The Graduate School of Arts and Sciences, University of Tokyo 3-8-1 Komaba, Tokyo 153-8902, Japan
jc_mc_datsu@yahoo.co.jp
[2] Graduate School of Science and Technology, Kobe University, Nada, Kobe 657-8501, Japan
tharuna@penguin.kobe-u.ac.jp

Abstract. In this paper, an expression of embodiment on computation and its effect are explored. By introducing the notion of latent state space, embodiment is formalized as an extension of state space and its projection process and resulting misidentification of states. Implementing this structure in the game of life, we investigate the relationships between the original system and misidentification of states by means of informational structures. As a result, we observe that, in particular case, misidentification affects the system's behavior and is relevant to maintain the property of the original system.

Keywords: Game of Life, Cellular Automata, Embodiment.

1 Introduction

The aim of this paper is to explore the concept of embodiment in the theoretical sense. Recently, the term "embodiment" is used in the wide range of fields and contexts but it is still a difficult concept to deal with. In robotics, introducing the embodied agent, lots of cognitive phenomena have been realized by the direct interaction with the environment, constructing its functionality (i.e. internal network architecture) in a bottom up manner [1]. The method suggests that our body regulates the way to interact with the environment and simultaneously shapes and determines our functionality to accompany a certain cognitive phenomena [2].

On the other hand, philosophers offer another aspect of body [3]. Let's remind the famous thought experiment proposed by Dennett [4]: Imagine a brain floating around in a vat of chemicals and nourishments, and kept stimulated by various electrodes that carry information about the world. Moreover, imagine that the technician had copied the brain's functional structure and all of the information in it to a computer program and it still accompanies experience and cognition. This type of understanding to cognition and experience constitutes the functionalist perspective. This means that as long as the information and the program was running appropriately, we need neither body nor even brain. In those cases, the necessity of the body is just to do something actually, to

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 75–82, 2011.

take action in some way in the world. Now, let's invert this story as follows. To maintain the function of the brain in the vat, we should keep filling up the chemicals and nourishments from outside. Even if the information and the functional structure in the brain was copied to the program, to run the software, the underlying hardware should be constantly powered by electricity. But those examples are only the part of the whole story. If we press forward this line of thought, pursuing the origin of the source constituting the body, a grounding point would never come and it would spread all over the environment. This is called the frame problem, in this case, the problem to define the constitution of the body. So we can say that the first story, the functionalist stance, would not require body or even brain in principle, and for the second story, body would be absorbed into the environment.

One important lesson that we can learn from these stories are that the concerning system which constitutes the consistent explanation is realized because of our appropriate selection of the frame. It means that the composer of the system is ignoring the underlying assumption and setting some limitation to their execution environment. But in the case of the real cognitive system, the execution environment seems not to be given in a fixed manner, but rather managing it on their own right and realizing their cognition through their body. That is, because of the body exposed in the constant connection with the environment and being unable fully to escape from its infection, the concerning system (or its functionality) would not be consistently expressed but rather defined in an *ad hoc* manner and functioning in the world. In this paper, we concern those double-barreled bodies that carry the role as an interface. Our aim is to understand the real cognitive phenomena by positively extending the concept of body in robotics. Dealing with those bodies, we could understand the primitive form of cognition, which is maintaining the identical functionality, and simultaneously openned up to the change in principle, such as development, learning, and adaptive behavior [5]. Delicate problem here is that the body of the concerning system won't be expressed by simply expanding its frame, the frame of the original system. This way of equipment would carry on exactly the same problem that we have wanted to solve. What we here have to express is the original system which is constantly exposed to the expansion of the frame. To express the formal structure of those conceptions, in this paper, we use cellular automata (CA) model. Particularly, to determine the behavioral modalities of the effect of embodiment, we use the famous CA model, the "Game of life" (GL) [6].

The GL is a well studied CA which has attracted much interest in the last decades. Defined as a two-dimensional square lattice, the model consists of an array of cells, each of which is in the state either off (0) or on (1) state. A cell interacts with the eight neighboring cells in accordance with a transition function, which is defined as follows: (i)If a cell is in state 0 and three of its neighbors are in state 1, then the cell's state becomes 1. (ii)If a cell is in state 1 and two or three of its neighbors are in state 1, then the cell's state remains 1. (iii)In all other cases, a cell's state becomes/remains 0. The transitions of cells take place simultaneously in discrete time steps. Many important characteristics of GL have been revealed

such as computational universality, a 1/f power spectrum, and self-organized criticality. Moreover, GL has been often used and chosen to demonstrate a particular biological characteristic because of its rich and complex behaviors. In this paper, we use GL to determine the relevance of embodiment to the system.

This paper is organized as follows. In the next section, we explain our motivation and how we express the embodiment in CA by introducing the notion of latent state space. In section 3, by implementing the latent state space in GL, we observe its behavior and analyze the informational structure it accompanies. Finally we give discussions and conclusion in section 4.

2 Expression of Embodiment Exemplified in CA

Let's consider a computing process on PC. Even in this case, we can find the body discussed in the previous section. Computation is a system which is mathematically defined and it corresponds to the information or the functional structure in the previous examples. This computation is connected to the environment through a device, a computer. Generally, various effects have been discussed, but one of the most famous example is the bit infection, constant growth of the undefined bits from the low order digit, and the resulting over flow of the computation. Although any computing process is latently exposed to the danger of miscalculation, they are well protected and limited by designers of computers. In this paper, we release this limitation in incremental steps, and observe the effect to the system. This could be considered as an example of embodiment on computation. In the next subsection, we will implement the analogy of this example into the computational process of CA.

2.1 Latent State Space: A Formal Expression of Embodiment

In order to express the above motivations, we here introduce the notion of latent space. Although it is originally introduced by using lattice theory [7,11], we avoid lattice theory in this paper. Let us consider a cellular state space $\Omega = \{0, 1\}$. Given a positive integer K, let Ω^K be the K-fold Cartesian product of Ω. Ω^K is called a *latent state space* relative to the state space Ω. Let $\pi_k : \Omega^K \to \Omega$ be a projection map to k-th coordinate ($k = 0, 1, \cdots, K-1$). That is, $\pi_k(x_0, \cdots, x_k, \cdots, x_{K-1}) = x_k$. π_k gives rise to an equivalence relation θ_k on Ω^K: $\mathbf{x}\theta_k\mathbf{y} \Leftrightarrow \pi_k(\mathbf{x}) = \pi_k(\mathbf{y})$. Since we have an isomorphism $\Omega^K/\theta_k \cong \Omega$, we have K alternative isomorphic representations for the state space Ω. This vagueness of the representation of Ω is the key to the following constructions.

Consider that two cells (numbered 0 and 1) are interacting with each other. They changes their states by following an intercellular interaction rule $f : \Omega^2 \to \Omega$ between them. The state of cell 0 is denoted by $w_0 \in \Omega$ and the state of cell 1 is denoted by $w_1 \in \Omega$. Suppose $K = 2$. In order to express vagueness of states $w_0, w_1 \in \Omega$, we shall associate an element $\mathbf{x}_i \in \Omega^2$ and a projection π_{k_i} such that $\pi_{k_i}(\mathbf{x}_i) = w_i$ with w_i for $i = 0, 1$. Each cell receives an input (w_0, w_1) to it as $(\mathbf{x}_0, \mathbf{x}_1) \in \Omega^2 \times \Omega^2$. Then it observes both $\mathbf{x}_0$ and $\mathbf{x}_1$ by a single projection π_0 or π_1 and make a identification of the input in order to follow the rule f. In this

setting misidentification of an input to a cell can occur. For example, consider the case $(w_0, w_1) = (0,0), \mathbf{x}_0 = (0,1), \mathbf{x}_1 = (1,0), k_0 = 0$ and $k_1 = 1$. If cell 0 choose π_0 then it identifies the original input $(0,0)$ as $(0,1)$. If cell 0 chooses π_1 then it identifies the original input $(0,0)$ as $(1,0)$. Thus, in this example, for any choise of projection the cell makes misidentification.

2.2 Model Description

In this subsection, we implement the above structure into GL. We define the data of a cell at site (i,j) and time t by the triplet $(\mathbf{x}^t_{i,j}, w^t_{i,j}, k^t_{i,j})$ with $\pi_{k^t_{i,j}}(\mathbf{x}^t_{i,j}) = w^t_{i,j}$, where $\mathbf{x}^t_{i,j} \in \Omega^K, w^t_{i,j} \in \Omega$ and $0 \leq k^t_{i,j} < K$. $\mathbf{x}^t_{i,j}$ is called a *latent state* of the cell. $w^t_{i,j}$ is called a *state* of the cell. The projection to the $k^t_{i,j}$-th coordinate $\pi_{k^t_{i,j}}$ is called an *observation*. Next we describe the rule dynamics in our formalism. Suppose that the transition function of GL is expressed as $f : \Omega^9 \to \Omega$. A cell at site (i,j) and time t receives an input $(w^t_{i-1,l}, w^t_{i,l}, w^t_{i+1,l})_{l=j-1,j,j+1} \in \Omega^9$ to it as a 9-tuple of latent states $(\mathbf{x}^t_{i-1,l}, \mathbf{x}^t_{i,l}, \mathbf{x}^t_{i+1,l})_{l=j-1,j,j+1}$. The number of possible observation is K. Since we external observers describe the behavior of cells by introducing the state space Ω, the degree of misidentification of an input should be minimized in the actual realized observation. Motivated by this consideration, we define the time evolution of cellular data as follows; define $\sigma^t_{i,j,k} := \sum_{m=i-1,i,i+1} \sum_{l=j-1,j,j+1} d(w^t_{m,l}, \pi_k(\mathbf{x}^t_{m,l}))$, $\delta^t_{i,j} := \min_{0 \leq k < K} \sigma^t_{i,j,k}$ and $S^t_{i,j} := \{k | \sigma^t_{i,j,k} = \delta^t_{i,j}\}$, where d is a metric on Ω given by setting $d(0,1) = 1$. The update of the data of a cell at site (i,j) and time t is performed by the following three steps.

(i)Choosing an observation. An element of $S^t_{i,j}$ is chosen by following the equiprobability distribution on $S^t_{i,j}$. We write the chosen element $k^{t+1}_{i,j}$.

(ii)Updating material factor $\mathbf{x}^t_{i,j}$. For $k \neq k^{t+1}_{i,j}$, $\pi_k(\mathbf{x}^{t+1}_{i,j}) = \pi_k(\mathbf{x}^t_{i,j})$. For $k = k^{t+1}_{i,j}$, $\pi_{k^{t+1}_{i,j}}(\mathbf{x}^{t+1}_{i,j}) = f((\pi_{k^{t+1}_{i,j}}(\mathbf{x}^t_{m,l})_{m=i-1,i,i+1,l=j-1,j,j+1})$. These equations determines $\mathbf{x}^{t+1}_{i,j}$ uniquely since $\mathbf{x}^{t+1}_{i,j} = (\pi_0(\mathbf{x}^{t+1}_{i,j}), \cdots, \pi_{K-1}(\mathbf{x}^{t+1}_{i,j}))$.

(iii)Updating formal state $w^t_{i,j}$. $w^{t+1}_{i,j} = \pi_{k^{t+1}_{i,j}}(\mathbf{x}^{t+1}_{i,j})$.

The above three steps defines the time evolution of the cellular data from $(\mathbf{x}^t_{i,j}, w^t_{i,j}, k^t_{i,j})$ to $(\mathbf{x}^{t+1}_{i,j}, w^{t+1}_{i,j}, k^{t+1}_{i,j})$. We call the time evolutionary system defined above "Embodied Game of Life" (EGL). Note that if $K = 1$ then any EGL is just an usual GL. In what follows, we only adopted the periodic boundary condition and the random initial condition to run the system. The random initial condition here means as follows. For each cell at site (i,j), first its initial state $w^0_{i,j}$ is chosen to be 0 or 1 with probability 0.5. Second its initial observation $\pi_{k^0_{i,j}}$ is chosen from the set of all observations $\{\pi_0, \cdots, \pi_{K-1}\}$ with equiprobability. Then a coordinate of its initial latent state $\pi_{k^0_{i,j}}(\mathbf{x}^0_{i,j})$ is automatically determined by $\pi_{k^0_{i,j}}(\mathbf{x}^0_{i,j}) := w^0_{i,j}$. Finally the value of $\pi_k(\mathbf{x}^0_{i,j})$ is chosen to be 0 or 1 with probability 0.5 for $k \neq k^0_{i,j}$. Thus an initial condition for a site is specified.

Moreover, if the value of $\delta^t_{i,j}$ is not 0, it means that the cell's identification of inputs contains mistake. Hence we call those cells *mistake cells*.

3 Behavior of EGL

In this section first we observe behaviors of some examples of EGL in order to find a clue of the effect of embodiment. Particularly, we concentrate on the behaviors of the system (*on-cells*, cells with state 1) and mistake cells. Second we analyze the information structure of the system and characterize the relevance of the embodiment.

3.1 An Observation

Fig.1 shows the time evolution of EGL and mistake cells. As time evolves, not just a pattern of EGL but also the one of mistake cells showed various behaviors. Especially, we observed that the mistake cells often interrupted the clouds generated by on-cells to be mixed each other. Moreover, we observed those cells stabilizing the pattern of on-cells to extra-ordinary ways. For example, in $K > 1, T = 1000$, various different fixed patterns were observed compared with GL ($K = 1$).

To observe the global property of the system, we performed a spectrum analysis. In [8], it is shown by spectral analysis that the GL displays $1/f$ noise. Fig.2(b) shows the results. We can see that, in $K = 2$, the original property of GL is maintained and otherwise they show a Lorentzian spectrum. In the next subsection, to understand how the mistake cells and the on-cells are related each other, we analyzed the informational structure among them.

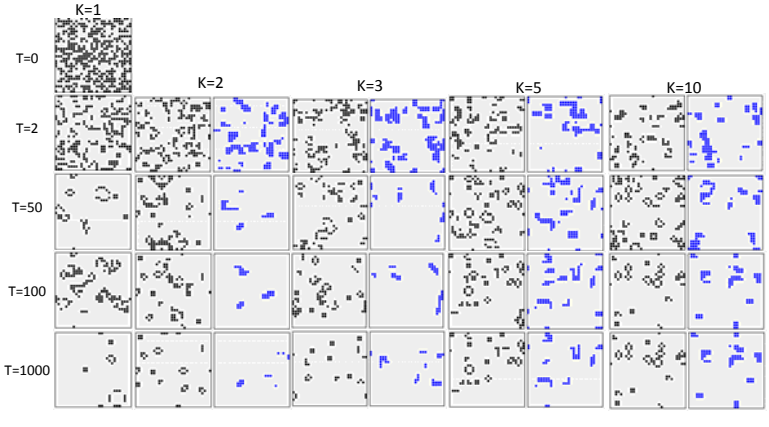


Fig. 1. Time evolution of a 30×30 random pattern for $K = 1, 2, 3, 5, 10$. For each diagram, right side represents the time evolution of mistake cells. All the diagrams are generated by setting the same initial pattern.

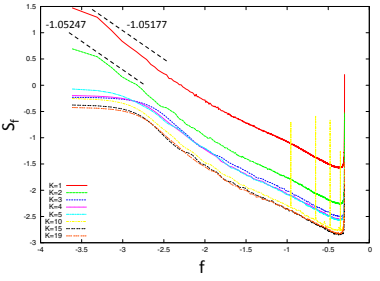


Fig. 2. Power spectrum at $K = 1, 2, 3, 4, 5, 10, 15$ and 19. We employed arrays with 100×100 cells and the number of time steps was set to 4096. In the case of $K = 1$ and $K = 2$, $1/f$ spectrum was obtained.

3.2 Relevance of Mistake in EGL

To quantify the properties of the space-time patterns of the system according to the change of K, we used a measure called *input entropy*, which is defined as follows: $E(t) = -\sum_{i=1}^{512} \left(\frac{Q_i(t)}{N} \times \log \frac{Q_i(t)}{N} \right)$, where N is a system size and $Q_i(t)$ is the number of the ith pattern consists of 9 bits in t. We calculated the input entropy of on-cells ($E_o(t)$) and mistake cells ($E_m(t)$) and observed their behavior. Fig.3(a) shows some typical examples of time series of input entropy. We can clearly see that as K grows, the value of the $E_o(t)$ remains almost the same but the time deflection of it gradually gets smaller. On the other hand, for the mistake cells, although the time deflection of the $E_m(t)$ remains almost the same, the mean value gradually gets smaller. These features could be extracted from Fig.3(b). The classification of space-time patterns can be described with respect to input entropy as follows [9]: (i) Ordered: Low averaged input entropy and low standard deviation. (ii) Complex: Medium averaged input entropy and high standard deviation. (iii) Chaotic: High averaged input entropy and low standard deviation. Since the GL is known to be complex, the results implicates that as K grows, the system gradually loses its complexity. On the other hand, the mistake cells are classified as ordered in the first place but especially in $K = 2$, its standard deviation is larger than others.

To pursue the relation between the on-cells and the mistake cells furthur, we used a measure that aims at extracting directed flow (transfer of information) between time series, called *transfer entropy*. Given two time series x_t and y_t, transfer entropy essentially quantifies the deviation from the generalized Markov process: $p(x_{t+1}|x_t) \approx p(x_{t+1}|x_t, y_t)$, where p denotes the transition probability. If the deviation from a generalized Markov process is small, then the state of Y can be assumed to have little relevance on the transition probabilities of system X. If the deviation is large, however, then the assumption of a Markov process is not valid. The incorrectness of the assumption can be expressed as follows: $T(Y \to X) = \sum_{x_{t+1}} \sum_{x_t} \sum_{y_t} p(x_{t+1}, x_t, y_t) \log \frac{p(x_{t+1}|x_t, y_t)}{p(x_{t+1}|x_t)}$, where the sums are over

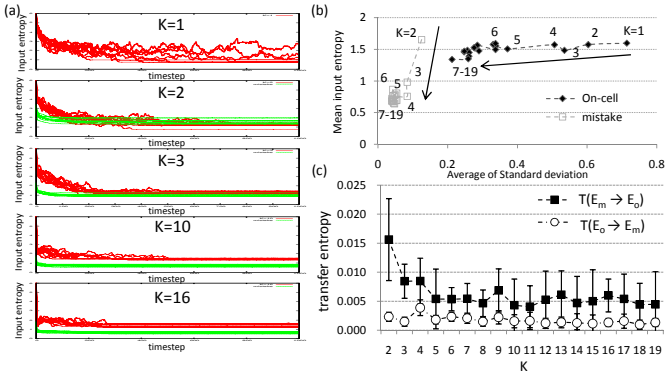


Fig. 3. (a)Time evolution of an input entropy of on-cells and mistake cells for $K = 1, 2, 3, 10$ and 16. Orbits are overlaid for 10 trials in each diagrams. (b)The mean of input entropy vs the averaged standard deviation of input entropy. Data of on-cells and mistake cells for $K = 1 - 19$ is plotted. Each dot is the average over 10 trials. For the mean of input entropy, calculating the average input entropy for each trial, we got the mean over those values. (c)The plots of transfer entropy according to K. Data of $T(E_o \to E_m)$ and $T(E_m \to E_o)$ for $K = 1 - 19$ is plotted. Each dot is the average over 10 trials. By discretizing the state into intervals which length 0.1 each, we calculated the transition probabilities for each trial and obtained the concerning value. We employed arrays with 50×50 cells and the number of time steps was set to 1000.

all amplitude states, and the index $T(Y \to X)$ indicates the influence of Y on X. The transfer entropy is explicitly nonsymmetric under the exchange of X and Y, and can thus be used to detect the directed exchange of information between two systems. The method is frequently applied in the field of sensorimotor coupling system research to quantify the informational structure over the redundant network architecture [10]. Since the input entropy reflects the complexity of patterns, we here analyze the information transfer between E_o and E_m (i.e. $T(E_o \to E_m)$ and $T(E_m \to E_o)$).

Fig.3(c) is the results. At first, we can see that the information transfer from E_m to E_o is larger than that from E_o to E_m in each case. And especially in $K = 2$, it is clear that the strength of $T(E_m \to E_o)$ is relatively larger than the others. On the other hand, $T(E_o \to E_m)$ remains almost the same value in each case. This would suggest that, in $K = 2$, the complexity of the patterns of mistake cells affect the dynamics of on-cells and is relevant to the system's maintenance of the original property of GL revealed in the previous subsection.

4 Discussions and Conclusion

In this paper, by introducing the notion of latent state space, we discussed the formalization which expresses the embodiment on computation, and implementing the construction to the GL, we examined the behavioral modality of the

system. In the previous work, the same formalization has been applied to the elementary CA and shown that it enhanced the class 4 behaviors [11]. Similar study could be also found in [12]. In those studies, it is said that the main driving mechanism of those behaviors is the expression of the original system which is exposed to the expansion of the frame. In this paper, we aimed to determine the informational structure between mistake cells (influences from outside of the frame) and the original system. As a result, we observed that, in particular case ($K = 2$), there exists a situation that the property of the original system is maintained by the influence from mistake cells. This could be an example of the system whose property is not constituted by the static function but rather maintained by the structural change of its functionality. On this point, further study is expected for the future work. Especially, by examining the behavior to the external perturbation, the system's adaptability or robustness would be precisely explored.

References

1. Brooks, R.A.: Intelligence without representation. Artificial Intelligence 47, 139–160 (1991)
2. Pfeifer, R., Scheier, C.: Understanding Intelligence. MIT Press, Cambridge (1999)
3. Gallagher, S., Zahavi, D.: The Phenomenological Mind: an introduction to philosophy of mind and cognitive science. Routledge, London (2008)
4. Hofstadter, D.R., Dennett, D.C.: The Mind's I: Fantasies and Reflections on Mind and Soul. Basic Books, New York (1981)
5. Nakajima, K.: Formalization of Embodied Sensorimotor Coupling System. In: Dubois, D. (ed.) AIP Conference Proc., Computing Anticipatory Systems, pp. 233–243 (2007)
6. Berlekamp, E.R., Conway, J.H., Guy, R.K.: Winning ways for your mathematical plays, vol. 2. Academic Press, London (1982)
7. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order, 2nd edn. Cambridge Univ. Press, Cambridge (2002)
8. Ninagawa, S., Yoneda, M., Hirose, S.: 1/f fluctuation in the "Game of Life". Physica D 118, 49–52 (1998)
9. Wuenshe, A.: Classifying Cellular Automata Automatically. Complexity 4, 47–66 (1999)
10. Lungarella, M., Sporns, O.: Mapping Information Flow in Sensorimotor Networks. PLOS Comput. Biol. 2, 1301–1312 (2006)
11. Haruna, T., Gunji, Y.-P.: A Protobiological Consideration on Cellular Automata. In: Proceedings of the 6th International Workshop on Emergent Synthesis, pp. 133–138 (2006)
12. Uragami, D., Gunji, Y.-P.: Lattice-driven Cellular Automata Implementing Local Semantics. Physica D 237, 187–197 (2008)

Metamorphosis and Artificial Development: An Abstract Approach to Functionality

Gunnar Tufte

The Norwegian University of Science and Technology
Department of Computer and Information Science
Sem Selandsvei 7-9, 7491 Trondheim, Norway
gunnart@idi.ntnu.no

Abstract. An artificial developmental process may reflect the principle of a process starting with a zygote which develops to a multicellular organism. An organism goes through an interwoven process of shaping the form and behaviour. Metamorphosis is a stage in the development of many species, e.g. insects, which include a large variation of phenotypic shape and behaviour in the life-time of the organism. Here principles from metamorphosis are included as a developmental stage that can be exploited by evolution to produce artificial organisms with variation in behaviour at different developmental stages. The target developmental system is a cellular system close to a non-uniform cellular automaton. As such, Darwin's discovery is exploited for evolving genomes for the construction (development) of von Neumann's cellular machines, Darwin meets von Neumann.

1 Introduction

Artificial developmental systems include system with some kind of mapping process that produce a phenotype out of the genotypic information by some kind of indirect iterative mapping process. The input information to the developmental process may include information from the adaptive process of evolution (the genome), environmental information (the conditions in which the organism develops) and intermediate phenotypic properties (cues provided by the emerging phenotype), —See [1].

In nature the developmental process is cellular, a process starting from a zygoth developing to an adult (multicellular) phenotype. A cellular process refer to communication, inter– and intracellular, autonomous processing of information in each cell, and that the cell is both the constructor and construct of the phenotype. The developmental process, the process of constructing the phenotype, may consists of stages, e.g. zygote, blastula, embryo, nymphs, larva, pupa, juvenile and adult, depending on the strategy adapted by the species, —See[2,3].

In living organisms functionality may be viewed as the purpose, i.e. reproduction, and the behaviour as the means of achieving the functionality. As such, the complexity of the organism, i.e. the behaviour, is given by the need to obtain an organism with a complexity necessary to achieve the purpose. In the artificial

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 83–90, 2011.

counter part a machine have a purpose and hence a functionality. The behaviour of the machine fulfils the purpose. The complexity of the machine most be at a level that makes the machine work, —See[4].

Taking the view of machines at a complexity level that makes them work into artificial development raises the question of what is needed to make (develop) a machine that work. Here this question is used as inspiration to look into, and trying to, include, inspiration from insect metamorphosis. Metamorphosis is an evolved biological adaptation implying stages. Stages in metamorphosis separate resources to deal with specialised tasks. The different tasks enables a kind of resource "optimisation" as resources can be aimed at obtaining intermediate organism properties on the developmental path to a reproductive adult [5].

Herein metamorphosis is taken as inspiration to reduce the resources usage in artificial development [6] and to enable artifacts with different behavioar given by life phases and/or external infuence. The stage vice development, including different phenotypic properties, is taken into the mapping process as stage where the resources used is relieved of functional requirements, i.e. fitness. The experimental approach shows how evolution and development (EvoDevo [7,3]) can exploit this stage's relive in pressure to be exploited for generating the general form of the "adult" phenotype from the non/different functional early phenotypic form.

2 Metamorphosis: Form and Function

As stated the process of development is an iterative process of creating and forming of the phenotypic structure. This implies an alteration of the phenotype during development, e.g. by growth, cell division and differentiation. The phenotype includes an inherent plasticity. This may be divided in two. Individual plasticity, i.e. phenotypic plasticity [8], an ability to adapt form and function of the developing phenotype depending on environmental conditions. As such, phenotypic form and function of artificial organisms can depend on environmental conditions as to be robust to environmental changes [9]. A second form of plasticity relates to the change of phenotypic form during development, i.e. developmental plasticity [10]. Metamorphosis is an evolved developmental plasticity [5] that enables development of phenotypes that include intermediate phenotypes with form and function that largely deviate from the adult (reproductive) phenotype.

In the view of evolution the origin of metamorphosis possible relates to exploitation of available resources by introducing a stage to transform one phenotypic form/function capable of exploiting available resources to a form for the actual function (purpose) of reproduction. The pupa stage in this transformation include an extreme expression of plasticity hence also a increased activation of genes. However, at this stage the functionality (or purpose) is not reproduction but the transformation to a functional (reproductive) phenotype.

By introducing stages in artificial development that is relieved of an actual functionality requirement, except for the emergence of a functional phenotype at a later stage in development, this intermediate stage can as in insect metamorphosis be concentrated on producing a form for a functional phenotype.

Introducing such a stage hopefully reduces the resources needed to develop a functional phenotype. This reduction in resources is here given as a time slot in development where there is no requirement of functionality. As such, this time slot in developmental time is exploitable (by evolution) to transform an early phenotypic form, e.g. larva, to an functional adult form.

3 EvoDevo: Darwin Meets von Neumann

In this work the structures targeted are developing structures capable of computation. The computational architecture is based on Cellular Automata (CA) originating from von Neumann [11]. von Neumann's Self-Reproducing Automata is in itself close to artificial development [12], cells with a finite number of states that can self-replicate, i.e. an expanding cellular structure. Herein the developmental phenotypes is based on a cellular computational machine [13]. A non-uniform CA is developed, i.e. the structure/form, emerges out of a set of developmental rules capable of cellular growth, differentiation and cell death. The developmental rules are evolved using a Genetic Algorithm (GA), a principle inspired by Darwin [14].

As stressed in Section 1 organisms have a functionality (purpose). The functionality here is the output of the non-uniform CA phenotype, i.e. an emerging behaviour given by running the CA.

Fig. 1 is an example of development of a cellular machine and it's behaviour. The phenotype is an emerging non-uniform Cellular Automata (top). Development of the structure goes through steps, Development Steps (DS), where the structure is formed by growth (expanding the number of cells) and differentiation (changing the rule of a given cell). The different colours in the emerging phenotype represent what CA rule the cell contains. White cells are considered

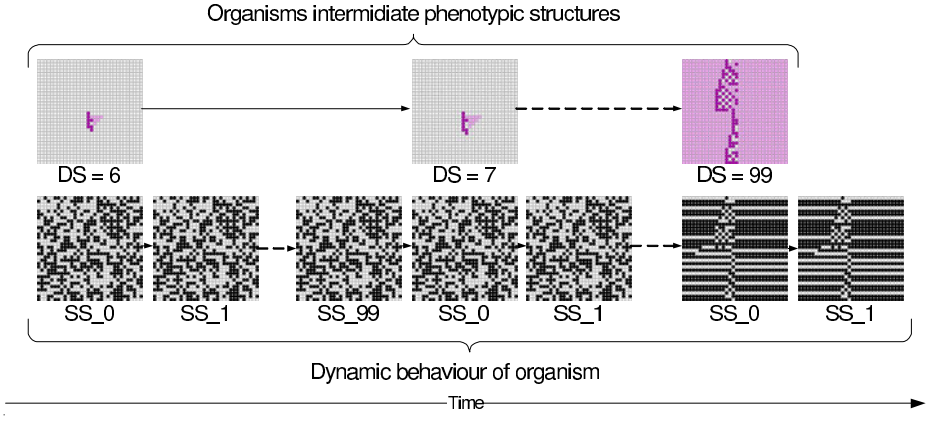


Fig. 1. Snapshot of the development of phenotypic structure (top) and the corresponding emergent behaviour (bottom) shown as space-time pattern

empty. The dashed lines indicate that there exists events that are not shown in the figure, e.g. the phenotypic structure between *DS 8* and *DS 98* are not shown. The behaviour of the system in Fig. 1 (bottom) is the state space produced from an initial state executed by the developing non-uniform CA. The space time plots for the behaviour consists of 100 State Steps (SS) for each development step. This implies that there exists 10 000 space time plots describing the behaviour of the system. It is important to note that in this system a behaviour exists from the first cell throughout the life-time of the organism. This opens for a adaptive system that can respond to externally enforced changes.

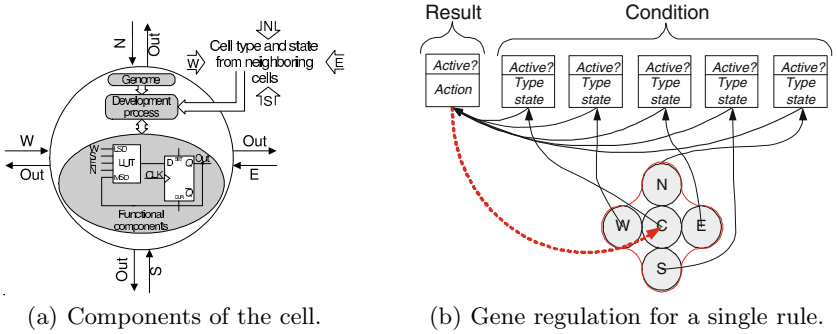
The evolution of a developing cellular machine in such a system exploits Darwin's discovery for evolving genomes for the construction (development) of von Neumann's cellular machines, Darwin meets von Neumann [15].

4 The EvoDevo System

The system as a whole is close to an Evolutionary Developmental (EvoDevo) approach —see e.g. [3,7]. This implies a developmental system with a possibility to include information from the environment, intermediate structures and behaviour in addition to the genetic information carried in the genome. The details of the system are only discussed in brief. For a complete description of the system —see [9] (developmental model) and [16] (evolutionary algorithm).

The development model is based on cellular development. This implies that the genome is present and processed autonomously in every cell. In the model, the cell also contains the functional building blocks. Fig 2(a) illustrates the developmental system — the cell. The cell is divided into three parts: the genome, development process and the functional component of the cell.

The genome consists of a set of rules. Rules are restricted to expressions consisting of the type and state of the target cell and the types and state of the cells in its von Neumann neighbourhood. There are two types of rules i.e. change and growth rules. Cell growth is a mechanism to expand the organism.



(a) Components of the cell. (b) Gene regulation for a single rule.

Fig. 2. The basic cell and a rule showing the gene regulation

Differentiation changes a cell's type i.e. its functionality. The result part of a change rule give the cell the target cell is going to be changed into. Cell death is a result of a change rule changing a cell type into empty whilst the state information is keept. As such, a cell can be killed off, but the state information in the cell is still avilable to neighbouring cells.

Each rule, shown in Fig. 2(b), consists of a result and a condition. The conditional part provides information about the cell itself (type and state) and each of the neighbouring cells. State information provides a way to include information relating to the functionality of the organism at a given point in time as well as information about the external environment — the empty cells in the environment also have state information. As such, a cell is represented in the condition of a rule by two genes representing its type and its state. However, a target cell is only represented by one gene: it's type for change rules or growth direction for growth rules. The state of cell may be 0, 1 or Don't Care (DC).

The functional components of the cell consists of a look-up table (LUT) defining functionality and a flip-flop as a memory element. The output value is synchronously updated and sent to all its four neighbours and as a feedback to itself. Available cell types were based on Sippers universal non-uniform CA [13] and threshold elements [17]. For further details on LUT definitions see [9].

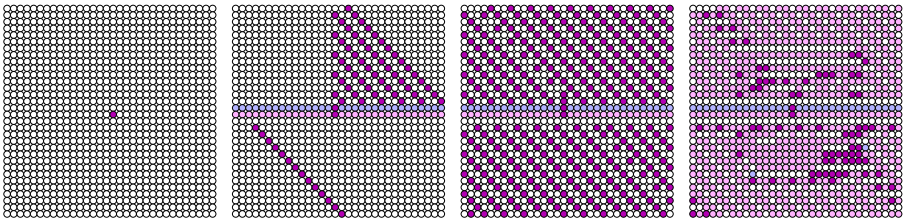
One update of the cell's type under the execution of the development process is termed a development step (DS). A development step is thus a synchronous update of all cells in the cellular array. The update of the cell's functional components i.e. one clock pulse on the flip-flop, is termed a state step (SS). A development step is thus made up of a number of state steps. An example of execution of developmental and state steps for the model can be seen in Fig. 1.

5 Experiment: Metamorphosis Enforced

In the previous section the principles of a EvoDevo system was described. In order to target the system to an abstract approach the metamorphic stage in the process of development was introduced by defining a portion of the available developmental steps differently and enforcing a portion of the cells to output a logical "1", a square three cell wide frame of 176 cells. As such, there is no internal regulation controlling developmental stages, rather defined enforced changes that can be exploited by evolution.

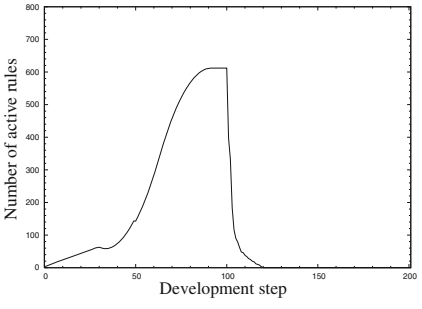
In the experiment the main behaviour (or functionality of the adult) is a sequential counter. Counting is based on the state information of the entire cellular space and the sequential operation of the functional components of the cells, i.e. the look-up table and flip-flop. A counting sequence is defined in the cellular array as the number of logical "1" in the cellular array increasing by one for each state step.

An organism witch goes through metamorphosis can in the early stage optimise its resources to reach an intermediate phenotype. Here this is represented as an entry in the fitness function counting the number of cell expressing a logical "1", i.e. a static pattern. In the metamorphic process there is no requirement

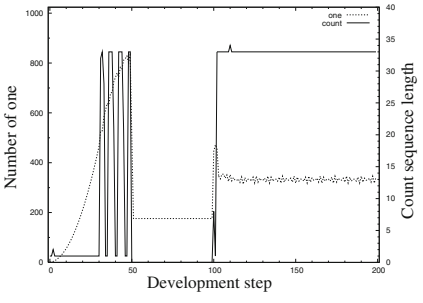


(a) DS 0 (b) DS 49 (c) DS 99 (d) DS 199

Fig. 3. The developing phenotypic structure at different development steps



(a) Number of cells expressing a active rule.



(b) Counter sequence and cells outputting a logical "1".

Fig. 4. Gene activation and lifetime behaviour

for functionality. The adult stage, the final targeted functionality, targets the counting behaviour described. As such, the organism goes through an initial developmental phase (larva) producing a static bit patter, the next step is the metamorphic transformation from bit patter generator to a functional counter behaviour. In the last phase (adult) the organism perform it's counter to the end of the apportioned number of development steps.

As stated, there is no functional requirement in the metamorphic stage. To further differentiate this stage to be exploitable by evolution each development step is here set to include zero state steps. As such, the change in state information between development steps is not present. This implies that the resources for a trajectory in the state space are reduced from 100 to 1 node for each development step.

The experiments was done using a genome consisting of 32 rules evolved for a maximum of 100 000 generations. As such, herein the target is to evolve developmental rules that can exploit the different stages as to produce a functional adult phenotype in the apportionated 200 DS, the functionality is given by 100 SS on each DS. The size of the cellular array was set to a maximum of 32 x 32 cells. In the experiment the state information of all empty cells ,except the initial zygote, was set to a logic '1'.

Fig. 3 shows an example of how the phenotypic structure develops. The initial configuration of the first single cell at *DS 0* is shown in Fig. 3(a). The intermediate phenotypic form and function (bit pattern) for the first stage of development is shown in Fig. 3(b). This stage is the final development step targeting a bit pattern functionality. From *DS 49* the organism goes through metamorphosis to a new phenotypic form shown in Fig 3(c). Between *DS 49* and *DS 99* there is no functional requirement to the developing organism. The phenotype shown in Fig 3(d) is the final phenotypic form considered.

The gene activation plot for the development of the organism in Fig. 3 is shown in Fig 4(a). The plot show the gene activation level, i.e. number of expressed rules in the organism, There are active rules, i.e. changes expressed in the phenotypic structure in the initial stage of development, *DS 0* to *DS 49*. In this phase the structure shown in Fig. 3(b) is the outcome of development.

In the Metamorphic stage, *DS 49* to *DS 99*, the gene activity is at its peek before decreasing when the metamorphic stage end. In the plot shown there is gene activity early in the adult stage finalizing the phenotypic structure and behavioar.

In Fig. 4(b) the counter sequence behaviour is plotted for the life-time of an individual together with the total number of cells outputing a logical "1". Here the metamorphosis is at its most prominent. At the early stage there are hardly any stable counter behaviour but the number of cells outputting a logical "1" encreases. This stage was not intended to include counter behaviour, it targets to maximize the number of logical "1" in the array. When the metamorphic stage is reached there are no counting behaviour at all, as the number of state steps for this stage is set to one. The absence of state steps provide a reduction in resources (state steps) and a more stable "environment" caused by an preservation of the regulatory input from the functional components of the cells. At *DS 99 - 100* the effect of the metamorphosis emerges as the adult behaviour emerges, a counter sequence of length 33. As such, if the behaviour is considered, it shows how the development creates a functional adult emerging at the end of the metamorphic stage.

6 Conclusion

In this work it is shown that metamorphosis can be included by defining a part of the available life-time of the organism as a specialised developmental stage. The introduction of such a stage can be exploited to create a phenotypic form that can produced functionality that may change during the life-time of the organism. Here the experiment shows example of how different functionalities can be targeted at different stages of development. In the example shown the metamorphic effect was clearly shown in form of gene activation and the change in behaviour throughout the life-time of the organism. In ongoing work the abstract approach are extended to evolve organisms that are sensible to external environmental variations as to produce different targeted functions. As such, the metamorphic stage is exploited as a transition phase for reshaping the phenotype to express different functionalities.

References

1. Kumar, S., Bentley, P.J. (eds.): On Growth, Form and Computers. Elsevier Limited, Oxford (2003)
2. Wolpert, L.: Principles of Development, 2nd edn. Oxford University Press, Oxford (2002)
3. Robert, J.S.: Embryology, Epigenesis and Evolution: Taking Development Seriously. Cambridge Studies in Philosophy and Biology. Cambridge University Press, Cambridge (2004)
4. Kampis, G., Gulyás, L.: Full body: The importance of the phenotype in evolution. Artificial Life 14(3), 375–386 (2008)
5. Truman, J.W., Riddiford, L.M.: The origins of insect metamorphosis. Nature 401, 375–386 (1999)
6. Tufte, G.: Phenotypic, developmental and computational resources: scaling in artificial development. In: GECCO 2008: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, pp. 859–866. ACM, New York (2008)
7. Hall, B.K., Pearson, R.D., Müller, G.B.: Environment, development, and Evolution Toward a Synthesis. The Vienna Series in Theoretical Biology. MIT Press, Cambridge (2004)
8. Larsen, E.W.: A View of Phenotypic Plasticity from Molecules to Morphogenisis. In: Environment, Development, and Evolution Toward a Synthesis, ch. 7, pp. 117–124. MIT Press, Cambridge (2004)
9. Tufte, G.: Evolution, development and environment toward adaptation through phenotypic plasticity and exploitation of external information. In: Bullock, S., Noble, J., Watson, R., Bedau, M.A. (eds.) Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems, pp. 624–631. MIT Press, Cambridge (2008)
10. West-Eberhard, M.J.: Developmental Plasticity and Evolution. Oxford University Press, Oxford (2003)
11. von Neumann, J.: Theory of Self-Reproducing Automata. University of Illinois Press, Urbana (1966)
12. Sipper, M., Tempesti, G., Mange, D., Sanchez, E.: Von neumann's legacy: Special issue on self-replication. Artificial Life 4(3) (1998)
13. Sipper, M.: Evolution of Parallel Cellular Machines The Cellular Programming Approach. Springer, Heidelberg (1997)
14. Darwin, C.: On the Origin of Species by Means of Natural Selection. John Murray, London (1859)
15. Kampis, G., Szathmáry, E. (eds.): Darwin Meets von Neumann, International Conference on the Simulation and Synthesis of Living Systems. Springer, Heidelberg (2009)
16. Tufte, G.: Cellular development: A search for functionality. In: Congress on Evolutionary Computation (CEC 2006), pp. 2669–2676. IEEE, Los Alamitos (2006)
17. Beiu, V., Yang, J.M., Quintana, L., Avedillo, M.J.: Vlsi implementations of threshold logic-a comprehensive survey. IEEE Transactions on Neural Networks 14(5), 1217–1243 (2003)

Local Ultrastability in a Real System Based on Programmable Springs

Santosh Manicka and Ezequiel A. Di Paolo

Center for Computational Neurosciences and Robotics,
University of Sussex, United Kingdom
santosh.manicka@gmail.com, ezequiel@sussex.ac.uk

Abstract. A way to move gradually towards an objective is by making sure at every step that there is as little deviation as possible while adapting to obstacles. This has inspired us to model a local strategy to eventually attain viability (equilibrium) in a real complex dynamical system, amidst perturbations, using ultrastability to make sure that the path to viability itself is viable. We have tested this approach on a real actuator powered by a technology called "programmable springs" that allows for real-time non-linear programmable actuation. Our experiment involves a problem in adaptation similar to the pole-balancing problem. To solve it, we use ultrastability in a novel way, looking at the viability of dynamical transitions of the system in its phase space, to tweak the local properties of the actuator. Observations show that our approach is indeed effective in producing adaptive behaviour although it still requires further testing in other platforms, thus supporting the original hypothesis that ultrastability can be an effective adaptive mechanism [3] and laying a foundation for a promising new perspective in ultrastable robotics.

Keywords: Ultrastability, programmable springs, dynamical systems.

1 Introduction

The study of adaptive behaviour is central to the study and synthesis of intelligence [1]. The field has seen numerous models of adaptation, a majority of them designed from a learner-perspective. They involve a process that helps the learning system adapt to the intricacies of the task at hand based on some kind of optimisation. These models attempt to minimize the difference in the achievable performance of the system and the ideal performance for a given task [2]. In general, adaptation as a process is problem-centric.

Ashby gave a different perspective to adaptation: to learn is to act with stability in the face of environmental challenges [3]. The challenge in this case is on the viability of the system itself. In Ashby's words, there are certain 'essential variables' in a brain-like networked system that must not exceed certain 'critical values' [3]. When the environment poses a challenge to the system and some of its essential variables exceed their critical values, viability is compromised. Ashby proposed a method called 'ultrastability' to deal with such situations. As essential variables move to critical values, an adaptive mechanism kicks in and tries new connection parameters for the network that defines the system. The mechanism remains active as long as the values remain

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 91–98, 2011.

critical, and stops acting once the essential variables are viable again. Ashby argued that the system in principle could find the new parameters through random search [3]. He implemented his idea in a 'Homeostat', an electromechanical device demonstrating various forms of learning [3]. It is worth noting that as a concept of adaptation, ultrastability is not performance-based, but it is based on the integrity of the system: its consequences for performance are implicit – an idea arguably closer to how adaptive mechanisms might work in natural organisms. After this breakthrough in the 1950's, only very few researchers have used ultrastability in synthetic adaptation [4]. Indeed, after the Homeostat, very few real ultrastable systems have been designed. Ultrastability can, in principle be applied to any complex dynamical system. One of our motivations is to test its generality by applying it to a different kind of electromechanical system.

A problem with practical applications of ultrastability is the efficiency of random search in large parameter spaces that characterises real complex dynamical systems, e.g., the mammalian brain. Ultrastability in such cases could take an impracticably long time to find the survival parameters. A solution to this problem could be to employ a 'controlled random' search which we describe here. The system used to test it is a real actuator powered by a novel technology called “programmable springs” developed at Sussex [5]. This technology employs a non-linear programming of the actuator's behaviour using a 'force-profile', a graph that indicates how much force should be exerted by the actuator under various conditions. In principle, they can be used to design the actuator to exhibit any complex rotary motion [5]. Programmable springs are a potentially rich technology for intelligent robot actuators as the profiles can be dynamically programmed, but one issue that needs to be addressed is that of the best method to program them adapted to specific needs. This is another motivation for our work. We describe an experiment using programmable springs for an adaptation problem where ultrastability, if used as proposed originally, might be impractical, time-wise. We implement a novel “controlled random search” flavour, wherein ultrastability is triggered *both* when the essential variables are off viability limits and their path towards viability is without a “viability structure” that we refer to as 'deviation'. When triggered, ultrastability mutates small 'pieces' of the force-profile that are deemed responsible for the deviated behaviour as observed in its real-time phase space, thus trying out new behaviour. It will also be argued that the proposed method can be generalized to other problems in adaptation as well.

2 Methods

Our experimental apparatus consists of a see-saw platform. Its rotary motion about the centre is controlled by a “programmable spring” (Fig.1). It is basically a rotary electromechanical actuator with angle-sensing capability and a control software that can exert a specific force and damping in a particular direction (clockwise or anti-clockwise) when the see-saw platform is inclined at a certain angle (negative, when the left end of the platform is lower than the right end and positive, otherwise). The net exerted force determines the angular velocity, which in combination with the angle of inclination forms the axes of the system's phase space. A 'force surface' over the actuator’s phase space for the profile in Fig.3 is depicted in Fig.2. The light grey areas indicate forces in the anti-clockwise direction and the dark grey areas indicate forces

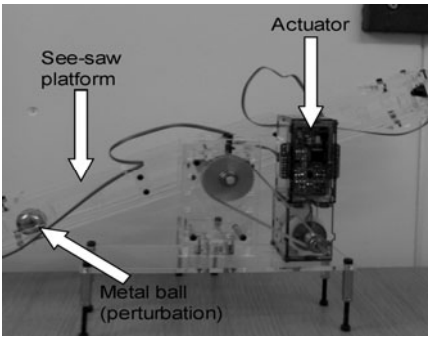


Fig. 1. The apparatus

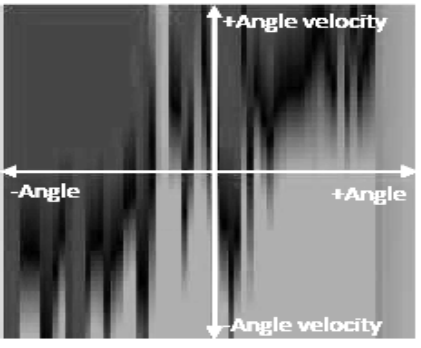


Fig. 2. Force surface for the profile in Fig.3

in the clockwise direction. The intensity of each shade indicates the intensity of the force exerted at that point. Black areas indicate zero force.

After the apparatus is switched on, the initial inclination of the platform and the shape of the force profile determine its ensuing behaviour. With a force surface symmetrical about the angle velocity axis and a uniform gradient along the angle axis, the platform reaches an 'equilibrium' point that corresponds to a horizontal inclination of the platform, regardless of where it is released from (Fig.4a). However, with the same profile, when a metal ball is introduced onto the platform, it reaches a point off the equilibrium point (Fig.4b). It can be seen that the perturbed behaviour finishes on the same side as the starting point simply because the underlying force profile is not robust enough to not give in to the weight of the ball (the light grey arrows in fig.4b show the same). In order to reach the equilibrium point in the latter case, one solution could be to restructure the force profile in such a way that the resulting behaviour looks similar to a symmetrical trail that naturally leads to stable equilibrium in the absence of perturbation (dark grey arrows in fig.4b). Thus by adopting a “natural” trail structure as a behavioural guide, a force profile that can adapt to perturbations could be found. Devising a method to accomplish this is the challenge addressed in this work.

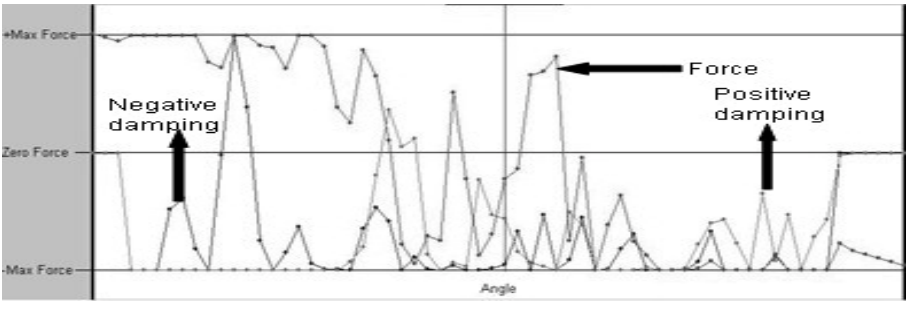


Fig. 3. A sample force profile. 'Positive damping' damps clock-wise rotation and 'negative damping' damps anti-clockwise rotation

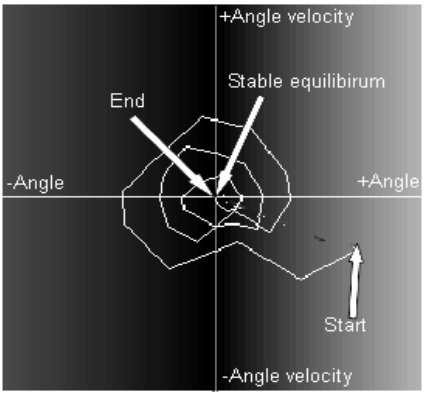


Fig. 4a. Actuator's behaviour without perturbation

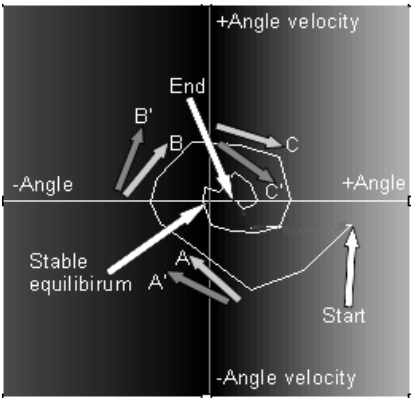


Fig. 4b. Actuator's behaviour with a permanent perturbation

In all the explanations that follow, the metal ball shall be considered a part of the apparatus. The proposed adaptation method consists of 3 steps: (1) actuation, (2) behaviour analysis and (3) profile mutation. Starting with a random profile, the actuator is activated and its trails in the phase space are captured for a predetermined length of time. The actuator is then paused and its behaviour during that period is analyzed by breaking the trail into individual 'transitions' like the light and dark grey arrows in Fig.5 below. In the analysis, we consider a random subset of these transitions with a predetermined number of samples (Parameter *V*). Any transition similar to the dark grey arrows (henceforth referred to as 'ideal transitions') are considered deviating with respect to a 'guide trail' as described in the previous paragraph, following which, the responsible tiny part of the profile is slightly mutated, that is randomly restructured in the spirit of ultrastability. The whole process is then repeated until a profile is found such that its trail spends a considerable length of time near the stable equilibrium point called as the 'Global Viability Zone' or 'GVZ' (the central square in Fig.5).

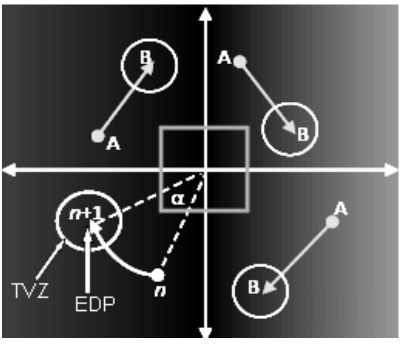


Fig. 5. A-B arrows are ideal transitions. The square is the GVZ and the circles are the TVZ's.

Each actual transition considered for analysis in a trail is compared with a corresponding ideal transition and if they are considerably different then the actual transition is considered deviating. An ideal transition at any point in the phase space is constructed as follows: if *n* is the starting point of a transition, draw an arc clock-wise from *n* with the stable equilibrium point (SEP) as the centre through a fixed angle α, the parameter *W* (Fig.5). The point *n+1* where the arc ends is referred to as the 'Expected Destination Point' (EDP). Then, a 'Transition Viability Zone' (TVZ) is constructed around the EDP whose radius is proportional (by a fixed 'TVZ radius factor', the parameter *X*) to the distance between *n* and EDP. If the actual destination point (ADP) in the actual transition is not within the TVZ, then the transition is deemed deviating. The EDP computation method captures the essence of approaching of the trail towards the SEP. The TVZ then acts as a "cushion" for the transitions so they don't have to be perfectly circular (in fact they can't be as they have to head in towards the SEP) and gives way to the formation of spiral trails. Again, the cushion lets spiral-like trails to emerge, thus giving room for complex behaviour. When a profile mutation is required because of a deviating transition, the responsible point on the force profile is looked-up and mutated as follows: if the absolute value of the angle velocity of the ADP is greater than that of EDP, then the absolute value of the force and damping values at that point are decreased and increased otherwise. The magnitude of mutation to be performed is computed as follows:

$$Magnitude = A \times B / C,$$

where, A = Random number in (0, Initial mutation range (Parameter *Z*)),

B = ADP-to-EDP distance,
C = Mutation range factor (Parameter *Y*).

The mutation range factor is manually set (an appropriate value was estimated by trial and error in our experiments). Mutations of proportionally lesser magnitude are also performed in the same direction on a few points near the chosen point in the profile in order to get a 'smooth effect', for the sake of gracefulness. After a predetermined initial mutation range is set, it is periodically tuned according to how well the adaptation is happening. After every few iterations of the actuation-analysis-mutation cycle, the current profile is scored and if it is better than the last evaluated profile, then the mutation range is narrowed a bit so the adaptations accumulated until then are more likely to be conserved, otherwise it is widened a bit so new adaptations could be explored. The method for scoring a profile is as follows:

$$Profile\ score = \frac{D - (2 \times E)}{F}$$

where, D = time spent by the trail in the GVZ,

E = standard deviation of the time spent in each quadrant of the phase space,
F = total time spent in the phase space.

As it can be seen, a number of control parameters are manually set before starting each experiment. A number of experiments were conducted by varying them. The following section will present the results from the most successful experiment.

3 Experiments

This experiment was run for about 10 hours at the end of which 842 profiles were generated. The following values were chosen for the parameters: V=250, W=30°, X=2, Y=80 and Z=100. Fig.6 below shows how the factors D, E (described above) and the profile score change with time. As it can be seen, there is a gradual improvement in factor A and the profile score over time. Since B seems to be almost constant, it can be assumed that A has more or less solely contributed to the improvement in profile score. The improvement in A can be attributed to the gradual approach of the average point in each quadrant visited by the evolving trails, towards the SEP (note the circles marked in Fig.7 where this approach has been captured). Fig.8 below shows the behaviour of the 600th profile that had the highest profile score of all. The trail indicates that it has developed some kind of 'one-arm strategy', with one active arm at a time (the two ellipses). Though it could give the indication that during these one-sided times, the ball is stuck on that side whereas in fact, it was not. This is supported by direct observation. Also, though not quite distinctly highlighted in fig.8, sudden drops in the angle velocity are also present. They characterise something we would like to call the 'braking behaviour' which actually let the ball roll smoothly on the platform thus performing a pause-and-pass act. Such an act in principle can enable the ball to gradually settle at the centre of the platform. Though it never fully successfully happened, the repeated braking behaviour indicates a trend to bring the ball to the centre.

Overall, the results show that the system develops a tendency towards reaching a dynamic equilibrium where the ball actuator behaves in such a way that the ball steadily sways about the centre of the platform and thus not simply succumbing to its weight. Experiments with other control parameter values resulted in different kinds of behaviours that could be in part explained by the choice of those values. Also, a slightly different approach to the mutation method was tried in it the direction of the mutation was also randomly chosen in a whole-hearted spirit of ultrastability, thus encouraging rapid exploration. The results (not presented here) showed emergence of interesting behaviours in a very short period of time. However, they quickly disappeared as the learned adaptations got easily eroded by the ensuing explorations.

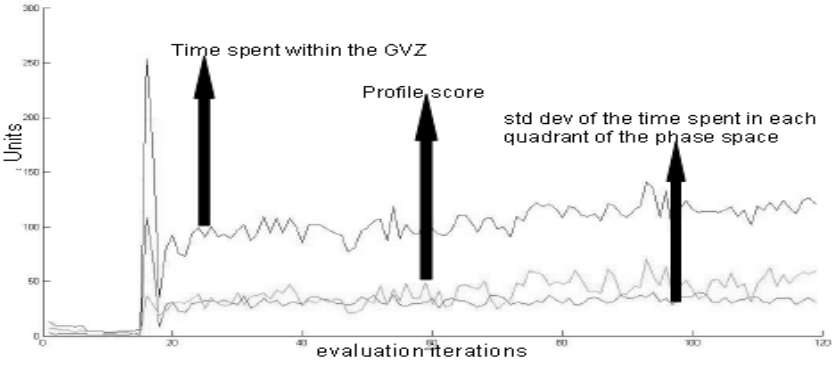


Fig. 6. Trends in profile score, time spent in the GVZ (A) and standard deviation of the time spent in each quadrant of the phase space (B)

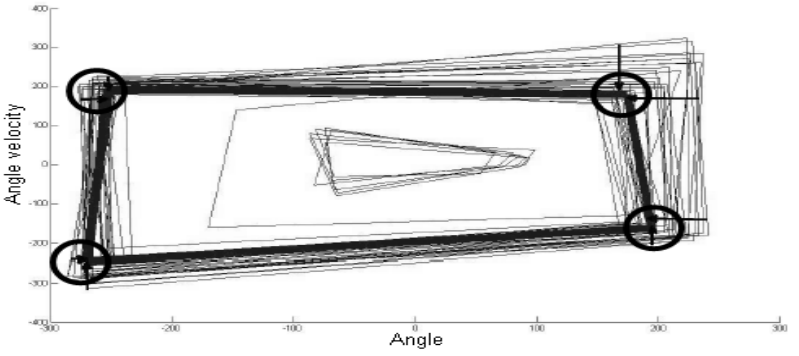


Fig. 7. A frontal view of the time evolution of the quadrant-wise average of the trails. The bold box is the latest average.

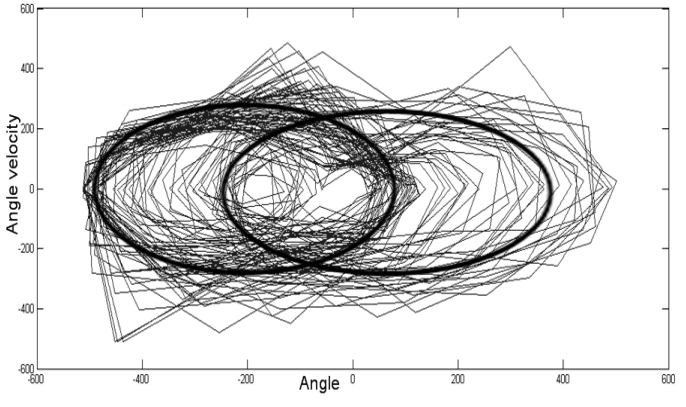


Fig. 8. Behaviour of a top scorer profile in the phase space. The bold ellipses mark approximately the average trajectories, each one slightly biased to one of the two sides of the platform.

4 Discussion

The objective of this work was to test ultrastability as an adaptive mechanism in a real system and to tailor it to suit the structure of the powering mechanism based on "programmable spring". Due to the continuous nature of the force profile and the need to adaptively shape the profile in a controlled fashion, the notion of viability zone in ultrastability was extended to the properties of the *dynamical transitions* in the phase space rather than defining it *only* for the state of the whole system. This is a novel conception of how ultrastability may be used. Our method was tested with a balancing problem. The ideal outcome of the experiments is a force profile adapted to the 'structure' of the perturbations (the dynamics of the moving ball) wherein the ball would be brought to a stand still in the middle of the platform, regardless of its initial position. With respect to this, the actual outcome could be treated as only partially successful.

One reason for this is the highly sensitive physical aspects of the system – even a slight movement of the ball off the centre can throw the system off balance and trigger behavior that keeps the ball in continuous motion. Repeated use also creates mechanical noise that cannot be easily removed and that the system has to adapt to further. Moreover, interesting behaviours might emerge if the EDP (Fig.5) were to be computed from a 'spiral-in' towards the SEP rather than from a circle as proposed in this work as the inward bias might encourage a stronger control around the SEP. Future research in this direction could take into consideration these design factors in order to achieve improved performance. With regard to the time to adaptation, our current approach takes quite a long time to find a suitable profile because the actuation, analysis and mutation steps happen one after the other. If they could be performed in parallel then the performance should in principle improve.

To summarize, it was shown that ultrastability can be used as an effective adaptive mechanism in a programmable spring based real electromechanical system. Our extended version of ultrastability can shape the dynamical phase space transitions of the system, moving beyond the conventional approach of monitoring only the state of the system. We believe that this approach can be used as an adaptive mechanism for other problems too with a proper, choice of the various parameters used in the method. For instance, if the actuator were to be used in a wheel that has to rotate continuously, then the SEP would be a line horizontal to the angle-axis depending on the desired velocity and the direction of rotation and the shape of a TVZ may be a rectangle inclined at various angles to the SEP line. We thus believe that the combination of a generic programmable spring technology and a generic adaptation mechanism based on ultrastability is a promising and workable tool for designing adaptable robots.

References

1. Beer, R.D.: Intelligence As Adaptive Behavior: An experiment in Computational Neuroethology. Academic Press Professional, Inc., San Diego (1990)
2. Landau, D.Y.: Adaptive Control: The Model Reference Approach. The “Control and System Theory” Series, vol. 8. Marcel Dekker, New York (1979)
3. Ashby, R.W.: Design for a brain: The origin of Adaptive Behaviour (2e). Chapman and Hall, London (1960)
4. Di Paolo, E.A.: Homeostatic adaptation to the inversion of the visual field and other sensorimotor disruptions. In: Meyer, J.A., Berthoz, A., Floreano, D., Roitblat, H., Wilson, S.W. (eds.) SAB 2000. From Animals to Animals, Proc. of the Sixth International Conference on the Simulation of Adaptive Behavior, pp. 440–449. MIT Press, Cambridge (2000)
5. Bigge, B., Harvey, I.: Programmable Springs: Developing actuators with programmable compliance for autonomous robots. Robotics and Autonomous Systems 55, 728–734 (2007)

Acquisition of Swimming Behavior on Artificial Creature in Virtual Water Environment

Keita Nakamura, Ikuo Suzuki, Masahito Yamamoto, and Masashi Furukawa

Graduate School of Information Science and Technology, Hokkaido University,
North 14, West 9, Sapporo 060-0814, Japan
{poco,ikuo,masahito,mach}@complex.eng.hokudai.ac.jp
http://autonomous.complex.eng.hokudai.ac.jp/

Abstract. The environment greatly influences acquirement of the behavior on the artificial life creature (AC) and artifact object (AO). However, the conventional studies like Karl Sims' ones have not accurately considered in an environmental influence. Instead, these influences are considered by replacing them into a simple environment. In this study, we accurately model the under-water environmental influence. And we propose a simulation method for artificial creature swimming in consideration of buoyancy and water drags as a virtual water environment. As a result of simulation, we verify that it is possible for AC and AO to acquire swimming behavior in the under-water environment. Additionally we show the analysis of the acquired swimming behavior.

Keywords: Physics modeling, Artificial Life, Animation, Simulation.

1 Introduction

A lot of simulations on the computer have been done for studying on acquisition of behaviors, evolution, and learning methodologies on a virtual artificial life creature and object. Terzopoulos et al.[1] realized a behavior of an artificial fish whose controller learns swimming in the virtual water environment. Sims[2] [3] showed that the virtual creature is able to acquire its morphology and behavior simulationaly by an evolutionary methodology based on the creature's competition. There have been a lot of studies based on Sims' studies. Chaumont et al.[4] applied Sims' model to evolution of virtual catapults. Miconi [5] observed coevolution of virtual creatures by fighting each other in Sims' virtual environment. In these studies, the experimental environment is set as an ideal environment in a computer simulation space. This is because they considered that the methodology of evolving learning behavior in an ideal environment is more important than acquisition of the similar behavior in a realistic environment. Therefore, the influence force from the environments to the virtual creature is not precisely analyzed. Instead, the implemented force adopted the simple calculation methods for reducing the computing time. On the other hand, in a field of numerical fluid dynamics, a lot of fluid simulations have been accurately investigated by a finite element method and a particle method. Koshizuka et al.[6] suggested the

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 99–106, 2011.

MPS method. They made it easy to create animation on the the water surface. Usami[7] did a simulation of swimming motion on Anomalocaris model in the virtual two-dimensional water environment using the particle method. But, the finite element method and the particle method require a great deal of computing time. So, it is unsuitable to do a real-time simulation for acquisition of behaviors in virtual environment using these methods. However, we insist that the virtual environment needs to obey the physical laws for the virtual creature to acquire a more natural policy of adaptive behaviors. In this study, we accurately model the under-water environmental influence and aim at doing simulation for the behavior of the virtual creatures in the realistic under-water environment. As a result of simulation, we verify that it is possible to acquire a swimming behavior for the virtual creatures in the virtual under-water environment.

2 Construction of Virtual Water Environment

We assume that the buoyancy and drag act as the force that a virtual object receive from the fluid effect. We construct a virtual under-water environment by modelling two forces acting on the object in the water. These two force compare to the buoyancy and drag, respectively(Fig.1). The simulation is performed by calculating movement of the object which obeys a physics law, resulting in an animation. We use the "PhysX (offered by the NVIDIA)"[8] as a physical calculating engine. PhysX is applied to calculate a basic physical operation, for example, gravity, and collision among the objects. In the virtual under-water environment, the density ρ of the water is 998.203[kg/m^3] and acceleration of the gravity g is 9.80665[m/s^2].

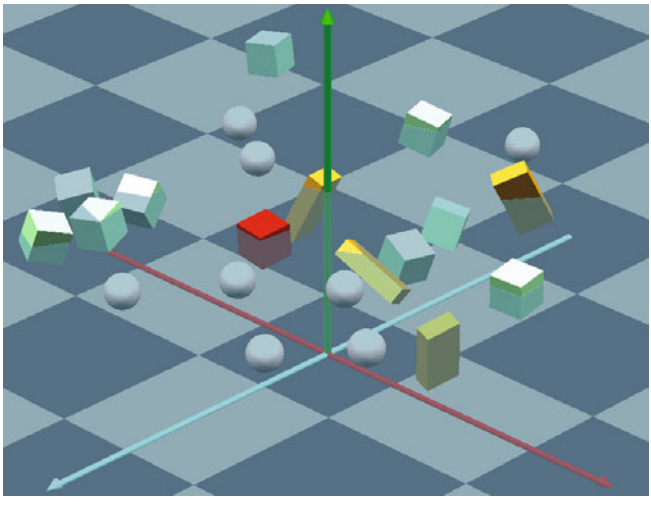


Fig. 1. Virtual Water Environment

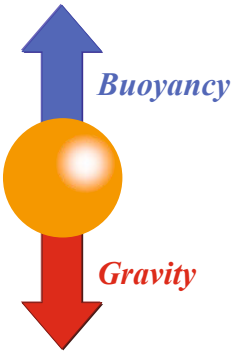


Fig. 2. Buoyancy Modeling

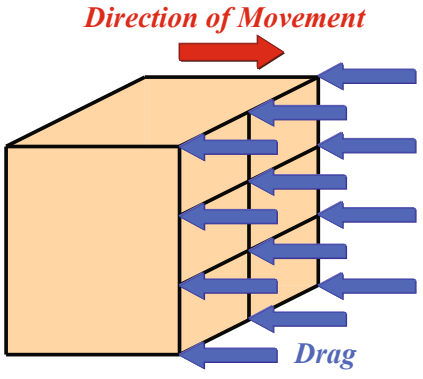


Fig. 3. Drag Modeling

Based on Archimedes' principle, we model the buoyancy as a force whose strength is equal to the weight of the water volume which an object occupied in the water. This force acts on the center of mass in the opposite direction of gravity(Fig.2). The strength of a buoyancy in the water, $F_{Buoyancy}$[N], is given by Eq.(1).

$$F_{Buoyancy} = \rho V g \tag{1}$$

where ρ[kg/m^3] is a density of the water, V[m^3] is a volume of the given object, and g[m/s^2] is an acceleration of the gravity.

We model the drag as uniformly distributed forces to the surface of the object(Fig.3). In a hydrodynamic field, using the dynamic pressure of a flow $\frac{1}{2}\rho U^2$[kg/(m$\cdot$s^2)] derived analytically as the strength of a drag in the water, the drag, F_{Drag}[N], is given by Eq.(2).

$$F_{Drag} = C_D \frac{1}{2}\rho U^2 S \tag{2}$$

where C_D is a scalar quantity called the drag coefficient, and S[m^2] is the reference area of the object. The drag coefficient depends on the shape of the object. In this study, the drag coefficient of a sphere is 0.47 and the drag coefficient of a rectangular solid is 1.50. The reference area of the object is the projection area of the object to the plane which is perpendicular to a flow.

3 Virtual Flounder Model

We examine how a virtual creature acquire adaptive swimming behaviors against the under-water resistance in our constructed virtual water environment. It is assumed that the creature must move towards the given light source as efficiently as possible. Evolutionary computing (EC) is adopted to obtain the adaptive behavior. This basic concept comes from Sims' proposed research. We create the

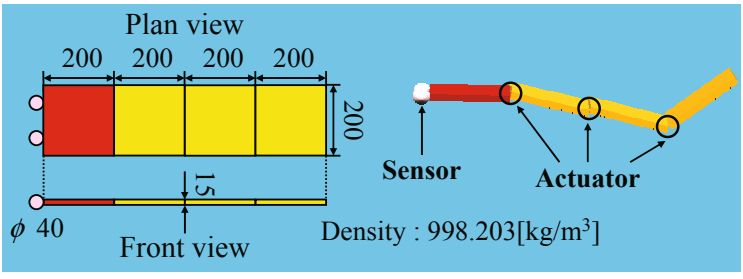


Fig. 4. Flounder1 Model

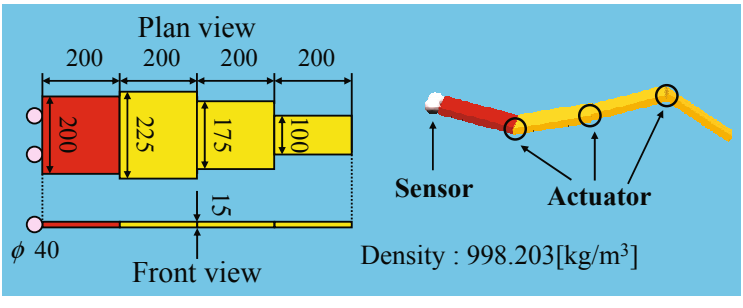


Fig. 5. Flounder2 Model

virtual creature by connecting a rigid body with actuators. The modeled virtual creature can act by controlling actuators. Two kinds of virtual creatures are prepared for this examination. Both of them imitate a rowing type of fish, which can swim by waving their body upside and down. After evaluation of the virtual creatures by EC, the obtained behavior is compared with that of an existing fish such as a flounder and a ray.

Fig.4 shows one virtual creature model, "Flounder1 Model", that we modeled. Flounder1 Model consists of four rectangular plates in the same size and two spheres in the same size as well. Their density is 998.2030[kg/m^3]. Fig.5 shows another model, "Flounder2 Model". Flounder2 Model consists of rectangular plates with different sizes and two spheres in the same size. Their density is the same as Flounder1 Model. These virtual creatures have two sensors at their eyes (spheres) , which can detect the angle between a light source and themselves and three actuators are set between rectangle plates(Fig.4 and Fig.5).

In this study, we control actuator sets implemented with the virtual creature. Actuators are controlled by outputs of the three-layer feed-forward artificial neural network (ANN). Table 1 shows the input and output parameters of ANN. We assume that each actuator generates an oscillating angular velocity in a sine function form. The output of i-th actuator ($i = 1, 2, 3$) is expressed by Eq.(3), where t is an elapsed time. ω_i is an angular velocity and ϕ_i is an initial phase delay. ANN controls (R and ω_i) in Eq.(3).

$$\omega_i = R\sin(\omega t + \phi_i) \tag{3}$$

The number of the neurons in the hidden layer is two times the number of the neurons in the input layer. Synaptic weights of ANN and an initial phase delay of each actuator are initialized by a random value at first. The virtual creature enables itself to swim towards a light source by optimizing ANN synaptic weights and the initial phase delay ϕ_i.

Table 1. Setting of Input and Output for ANN

Input	Relative angle of actuator i between plates in each time (θ_{xi}, θ_{yi}, θ_{zi})
	Relative actuator i angular velocity between plates in each time (ω_{xi}, ω_{yi}, ω_{zi})
	Initial phase that is decided by actuator i beforehand (ϕ_i)
	Sine and cosine of the angle θ_α for front projection between light source and sensor j in each time ($\sin\theta_\alpha$, $\cos\theta_\alpha$)
	Sine and cosine of the angle θ_β for plane projection between light source and sensor j in each time ($\sin\theta_\beta$, $\cos\theta_\beta$)
Output	Relative actuator i ideal angular velocity in each time (R, ω)

Table 2. Experimental Conditions

ANN	The number of the neuron of the input layer	29
	The number of the neuron of the hidden layer	58
	The number of the neuron of the output layer	6
GA	Genotype	W_{ij}
	Phenotype	d
	Population	20
	1 Step	1/60[sec]
	Simulation Step	900
	Generation	500
	Crossover Probability	0.9
	Mutation Probability	0.05

4 Experiments

We carry out experiments to examine how the virtual creature can acquire swimming behaviors towards a light source, and analyze the acquired swimming behaviors. We optimize the synaptic weights of ANN and the initial phase delay of each actuator using a real number type of the genetic algorithm (GA). Table 2 shows experimental conditions.

An evaluated value for GA as a fitness function F_{eval} is an accumulated distance d between the position of the sensor and the position of light source during swimming simulation steps in each generation given by Eq.(4).

$$F_{eval} = \sum_{t=0}^{Step} \sum_{j=0}^{s\_num} |\mathbf{x}_L - \mathbf{x}_{tj}| \tag{4}$$

where *Step* is the number of steps used for the swimming simulation at each generation, S_num is the number of sensors, $\mathbf{x}_L$ is the position of the light source, and $\mathbf{x}_{tj}$ is the position of the sensor j at each simulation step t. We use a rank selection as a reproduction operation based on the evaluated value and an elite preserving operation in GA. We sort the individuals in ascending order of their evaluated value and preserve the best five individuals. The others are modified by crossover and mutation operations. The virtual creatures evolve, learn, and acquire the swimming behaviors in the under-water resistance towards the light source by minimizing this evaluated value.

5 Results

Fig.6 shows a snapshot of an example of swimming behaviors that the 500th generation elite individual has acquired, drawn in every 150 step. Fig.7 shows the relation of the generation number and the mean of evaluated values in five times experiments on two models. Fig.8 shows outputs of actuators between 0-100 steps

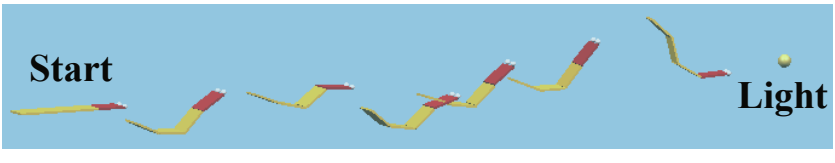


Fig. 6. A Snapshot of One Example of acquired swimming behavior

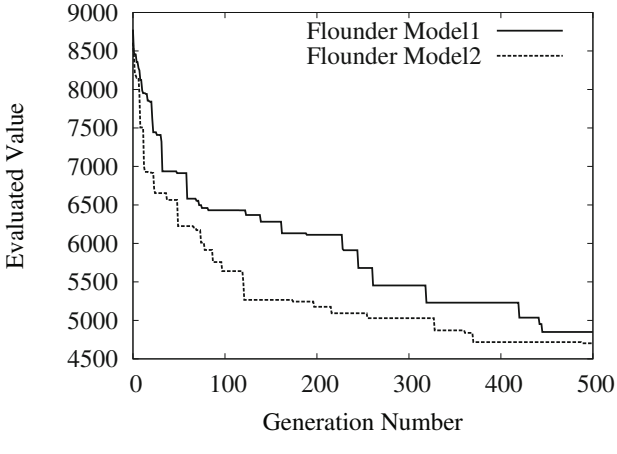


Fig. 7. Relation of the Generation Number and Evaluated Value

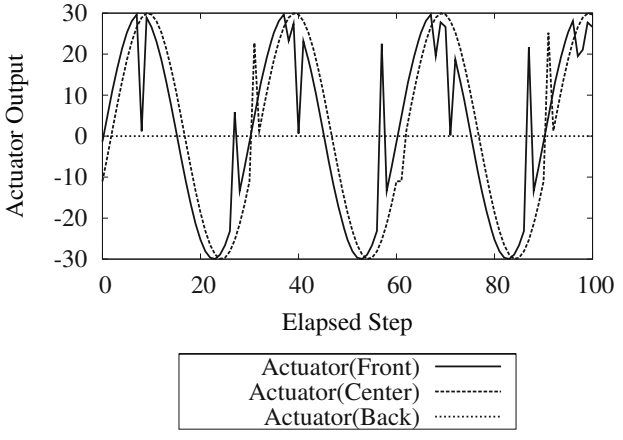


Fig. 8. Three Actuators' Output (Flounder1 Model)

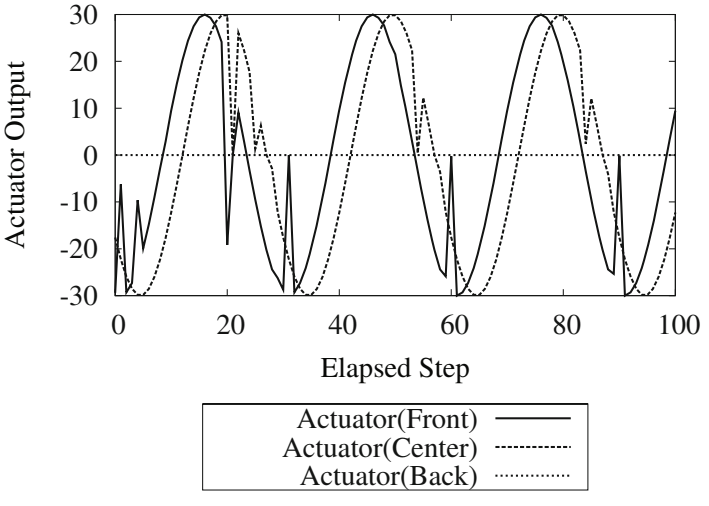


Fig. 9. Three Actuators' Output (Flounder2 Model)

after learning in "Flounder1 Model". Fig.9 shows them after learning in "Flounder2 Model". We upload the movies to URL[9] that each model acquired as the swimming behavior. From a result of Fig.7, as the number of the generations proceeds, both models decrease the evaluation values. Flounder2 Model resembling an existing fish in a shape, Then Flounder1 Model decreases the evaluation value faster. From results of Fig.8 and Fig.9, we can observe the following; (1) the actuator equipped with a head (front) plate generates the propulsive force leading to others, (2) the actuator equipped with a body (center) plate generates the propulsive force following the front one slightly late, (3) the actuator equipped with the tail (back) generates no propulsive force. This means the angular velocity

propagates from the front to the back. From these result, we prove that it is possible for the virtual creature to acquire the swimming behavior like a real fish in the rowing types under the water environment by using GA.

6 Conclusion

In this paper, we constructed the virtual water environment by introducing two forces comparing to buoyancy and drag by use of the physically calculating engine. And we performed experiments that the virtual creature acquires swimming behaviors in the constructed environment. As a result of experiments, we show that it is possible for the virtual creature to acquire the swimming behaviors under the water resistance. In addition, we analyzed the acquired swimming behavior.

In a future, we would like to experiment that virtual creature acquires swimming behaviors avoiding collision with obstacles in our constructed virtual water environment. We would like quantitatively to analyze the acquired swimming behaviors and verify the mechanism of the acquired swimming behavior. Furthermore, we would like to develop a general topology expression for presenting more sophisticated and various types of virtual creatures and explore "Life as it could be".

References

1. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes with autonomous locomotion, perception, behavior, and learning in a simulated physical world. In: Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp. 17–27. MIT Press, Cambridge (1994)
2. Sims, K.: Evolving virtual creatures. In: SIGGRAPH 1994: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, pp. 15–22. ACM, New York (1994)
3. Sims, K.: Evolving 3D morphology and behavior by competition. In: Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp. 28–39. MIT Press, Cambridge (1994)
4. Chaumont, N., Egli, R., Adami, C.: Evolving Virtual Creatures and Catapults. Artificial Life 13(2), 139–157 (2007)
5. Miconi, T.: In Silicon No One Can Hear You Scream: Evolving Fighting Creatures. In: O'Neill, M., Vanneschi, L., Gustafson, S., Esparcia Alcázar, A.I., De Falco, I., Della Cioppa, A., Tarantino, E. (eds.) EuroGP 2008. LNCS, vol. 4971, pp. 25–36. Springer, Heidelberg (2008)
6. Koshizuka, S., Nobe, A., Oka, Y.: Numerical analysis of breaking waves using the moving particle semi-implicit method. International Journal for Numerical Methods in Fluids 26(7) (1998)
7. Usami, Y.: Re-examination of Swimming Motion of Virtually Evolved Creature Based on Fluid Dynamics. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 183–192. Springer, Heidelberg (2007)
8. Nvidia physx, http://www.nvidia.com/object/nvidia_physx.html
9. Movie acquired swimming behavior, http://autonomous.complex.eng.hokudai.ac.jp/researches/physics-modeling/movies/nakamura/

Evolving Amphibian Behavior in Complex Environment

Kenji Iwadate, Ikuo Suzuki, Masahito Yamamoto, and Masashi Furukawa

Hokkaido University, Laboratory of Autonomous System Engineering,
9 Nishi, 14 Kita, Kita-ku, Sapporo, 060-0814 Japan

Abstract. In this study, we aim to evolving autonomous virtual creatures which have complex shapes in a complex environment. We implement a basic physics law and fluid influences with a virtual environment and evolve artificial creatures in plural environments (on the ground, in the water). Each model, which is evolved in a different environment, obtains an effective moving behavior in each environment (looks like walking, and swimming).

Keywords: Physics Modeling, Artificial Neural Network, Genetic Algorithm, Computer Graphics.

1 Introduction

The computer-aided animation using the computer graphics (CG) technology becomes more important in various fields such as physics, engineering, entertainment, and medical science. In order to create the realistic object motion, Physics modeling (PM)-based animation has attracted much attention to researchers and many researches have been presented in this decade. Well-known works are "smoothed particle hydrodynamics (SPH)"[1], and "moving-particle semi-implicit method (MPS)"[2]. These two methods are used for simulating and animating the fluid motion (water, flame, and smoke) in a field of production engineering, movies, and game industries[3]-[8]. On the other hand, the PM-based animation for artificial-beings, which can autonomously behave as exiting organisms in the earth, is still undergoing research matter. A motion capture method is mostly adopted to create the animation for the artificial-beings. However, this method consumes lots of time and requires a expert knowledge. One approach is the agent-based animation to overcome problems that the motion capture method has. The agent in a virtual environment can autonomously behave itself under restriction of physics lows. Such an agent is realized by equipping it with sensors, controllers, actuators, in physics modeling. We can also let the agent to evolve by giving some tasks.

This study aims at establishing a new computer aided animation method using the agent-based and PM-based animation. The specific problem we treat in this paper is to acquire an adaptive amphibian behavior in a complex environment and to automatically create the animation of its behavior. The amphibian is regarded as the agent. Therefore, we provide it with sensors, controllers and actuators. An artificial neural network (ANN) is used for the controllers. Evolution of the amphibian is realized by ANN learning. Evolutionary computation (EC) is introduced into ANN learning. The most different aspect comparing with conventional researches is to deal with a

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 107–114, 2011.

complex environment under restriction of the physics laws. We implement the Newtonian dynamics (ND) and fluid influences (FI) with the environment. The amphibian behavior must obey ND and FI. Forces against the amphibian, coming from ND and FI, are figured out by adopting PM with ease. Numerical experiments prove that we can obtain the adaptive behaviors, walking and swimming, for the amphibian in the complex environment and automatically create its behavior animation simultaneously.

2 Construction of Virtual Environment

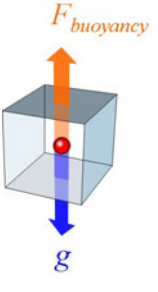
When we detail with our constructed virtual environment, all experiments described below is performed under this environment.

We use a dynamics engine for implementing the basic physics law with our constructed environment. The adopted engine is PhysX, presented by NVIDIA[9]. PhysX allows us to simulate physical dynamics and phenomenon such as rigid-body dynamics, elastic-body dynamics, fluid dynamics by using particle physics, and collision detection. It consumes much time to compute the fluid simulation by using the particle physics, since all particles motion must be computed. Therefore it is unsuitable for our approach because one of our goal is to make the real-time animation, which needs to compute physical dynamics repeatedly. Therefore, we regard fluid influenced forces as external forces. These forces act on objects instead of the forces coming from environment.

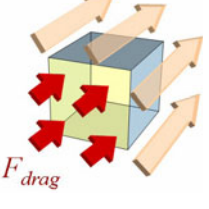
We introduce two forces caused by fluid. One is a buoyancy, and the other is a drag force. The buoyancy is given by Eq.(1). In the fluid the buoyancy artificially generated acts on the center of mass of the floating object (see Fig.1(a)). The Drag force affected by the fluid is given by Eq.(2). This drag force acts on points randomly selected on the surface of the floating object (see Fig.1(b)).

$$F_{buoyancy} = \rho V g \tag{1}$$

$$F_{drag} = C_d \frac{1}{2} \rho A v^2 \tag{2}$$



(a) buoyancy.



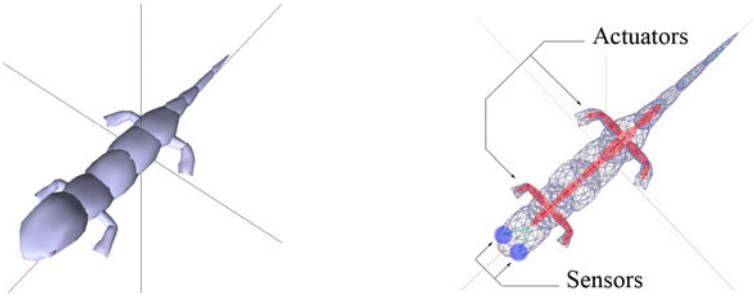
(b) drag force.

Fig. 1. Influence forces caused by fluid

3 Virtual Creature

We have developed a tool, which is capable of modeling a virtual creature and simulating it interactively. Two fluid forces are implemented with this tool as environment forces. We model a salamander using this tool. The salamander can behave itself under the water and air resistances, namely a complex environment. Then, we examine the salamander behavior by evolving it in the complex environment.

Our salamander consists of twenty rigid solid objects whose geometric and physical data are shown in Table 1.



(a) Figure of the salamander model. (b) Sensors and actuators.

Fig. 2. Salamander model

Table 1. Data of components of the salamander model

Body	**Value**
Density	$\rho = 1000$
Restitution coefficient	$\varepsilon = 0.1$
Static friction	$\mu_s = 0.5$
Dynamic friction	$\mu_d = 0.4$
Limbs	**Value**
Density	$\rho = 1200$
Restitution coefficient	$\varepsilon = 0.1$
Static friction	$\mu_s = 0.9$
Dynamic friction	$\mu_d = 0.8$

The salamander has two optical sensors and thirteen actuators (see Fig.2(b)). The optical sensors detect a light strength L from a light source placed in a virtual space. L is defined by Eq.(3), where θ_{Sensor} is an angle between the light source and the sensor direction and R is an angular range of the sensor. This value becomes an input of the salamander controller. The controller is modeled by an artificial neural network (ANN). A detail of ANN is explained in the next section.

$$L = \begin{cases} \theta_{Sensor} \le R & : \cos(\pi\theta_{Sensor}/R) \\ otherwise & : 0 \end{cases} \tag{3}$$

4 Artificial Neural Network (ANN)

ANN is a well-known brain model. It consists of a set of neurons (units) and set of synapses (arcs). Learning is performed by adjusting a set of weights assigned to arcs. A single unit is connected mutually and it has a number of inputs and outputs (see Fig.3). The input signal of each unit is defined by Eq.(4). The output signal of each unit is defined by Eq.(5).

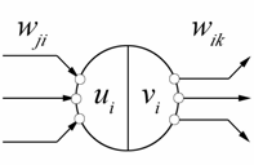


Fig. 3. Single unit model

$$u_i = \sum_j w_{ji} v_j \tag{4}$$

$$v_i = \left(1 + e^{-u_i/\tau}\right)^{-1} \tag{5}$$

u_i : input value of neuron

v_i : output value of neuron

w_{ji} : synaptic weight

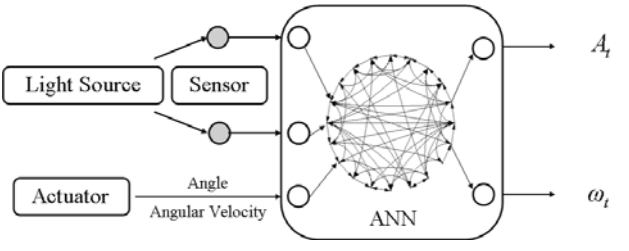


Fig. 4. Structure of ANN

Table 2. Components of the salamander

From each sensor	**Value**
Light strength	$L \in$ [0, 1]
Environmental pressure	$\sigma \in$ [0, 1]
Cosine of angle between sensors and vertical axis	$\cos(\theta_{Vertical}) \in$ [-1, 1]
Sine of angle between sensors and vertical axis	$\sin(\theta_{Vertical}) \in$ [-1, 1]
From each actuator	**Value**
Cosine of angle of each degree of freedom	$\cos(\theta_{Actuator}) \in$ [-1, 1]
Sine of angle of each degree of freedom	$\sin(\theta_{Actuator}) \in$ [-1, 1]
Cosine of a phase of the added torque	$\cos(\theta_{t-1}) \in$ [-1, 1]
Sine of a phase of the added torque	$\sin(\theta_{t-1}) \in$ [-1, 1]
Normalized gain of the torque	$\frac{A_{t-1}}{A_{\max}} \in$ [0, 1]

Fig.4 shows a structure of ANN we adopt. ANN consists of an input layer, an output layer and a hidden layer. The arcs of the Input layer are connected with sensors of

a salamander. They receive signals from sensors. The arcs of the output layer are connected with inputs of an actuator. The actuator set in the salamander converts signals given by its output into driving torques. These torques T_t given by Eq.(6), where θ_{t+1} given by Eq.(7) is a angle to determine the following step torque.

$$T_t = A_t \sin(\theta_t + \phi) \tag{6}$$

$$\theta_{t+1} = \theta_t + \omega_t \Delta t \tag{7}$$

The detail of sensors and actuators used in the salamander is shown in table 2.

5 Optimization of ANN

Since ANN controls actuators, a salamanders behavior is dominated by ANN. Therefore, adapting the salamander to an environment depends on adjustment of weights assigned to arcs in ANN. This adjustment is so called learning. However, it is difficult to define a learning signal to train ANN when the virtual creature has complicated shapes and it virtually lives in the complex environment. For learning, we adopt Genetic Algorithm (GA) with real number encoding. GA optimizes weights to acquire adaptive behaviors. A chromosome is represented by a set of weights. Table 3 shows parameters for optimization.

Table 3. Parameters of the optimization

Population of chromosome	$N_p = 20$
Crossover probability	$P_{Crossover} = 0.4$
Mutation rate	$P_{Mutation} = 0.01$
Simulation time	T = 2000 [step] (33.3 [sec])

The optimization process consists of the following steps.

1. Simulate each virtual creature in a constructed environment according to ANN generated by each chromosome.
2. Evaluate behaviors of each creature by use of a given fitness function.
3. Reproduce new creatures.
4. Perform GA operations (selection, crossover, and mutation).
5. Return to step 2 and repeat until the termination condition is satisfied.

The mutation operation randomly chooses a chromosome (a set of weights) and extract a gene (a weight) from it with the mutation rate $P_{Mutation}$ and replace the gene to a random number [-1.0, 1.0]. The crossover operation chooses a couple of chromosomes with the crossover probability P_C and selects one of the output unit and its neighborhood arcs (which are connected to the selected unit directly). Then these selected arcs of one chromosome are swapped to these of other chromosome.

We treat a learning problem what behavior a salamander can acquire when it moves towards a given goal. It is expected that walking, running, or swimming

emerges from the acquired behavior in the given environment. In this problem, a light source is set as the goal and salamander sensors detect the light (Fig.5). Since actuators equipped with the salamander is controlled by output layer signals of ANN, weights assigned to arcs in ANN are optimized such as the salamander moves towards the light source. As described above, GA is applied to optimize the weights. A fitness function, which takes consideration in three fundamental behavior evaluations, is described below.

1. Move from a current place to a goal via a path as short as possible.
2. Consume energy as efficient as possible.
3. A goal is always observed in the center of sight.

The first evaluation is formulated so as to minimize the cumulated distance D between the salamander's current place and the light source during the salamander moving. The second one is formulated by minimizing the consumed energy E , which is measured by cumulated torques generated by actuators. The third one is formulated so as to maximize the cumulated cosine value S of the angle between the sensor and the light source during the salamander moving. These three evaluations are expressed in Eqs. (9)-(11)

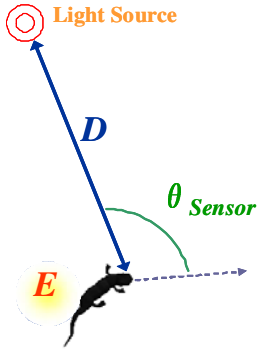


Fig. 5. Light source and evaluated values

$$D = \sum_t \left(\frac{\sum_i^{N_S} \left| \vec{P}_{Light} - \vec{P}_{i,t} \right|}{N_S} \right) \tag{9}$$

$$E = \sum_t \left(\sum_j^{N_A} \left| T_{t,j} \right| \right) \tag{10}$$

$$S = \sum_t \left(\frac{\sum_k^{N_S} \cos(\theta_{Sensor,t,k})}{N_S} \right) \tag{11}$$

$\vec{P}_{Light}$: Position of the Light source

$\vec{P}_{t,i}$: Position of sensors

N_S : Number of sensors

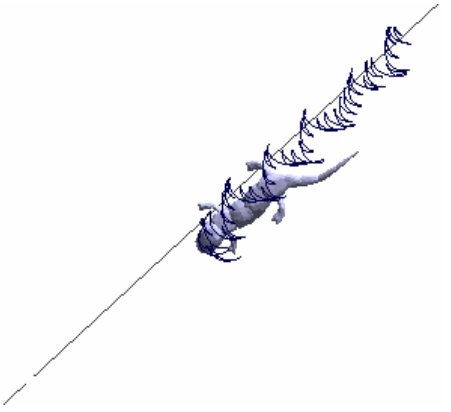
N_A : Number of actuators

As a result, the fitness function is expressed as a summation of Eqs. (9)-(11) with weights C_1 , C_2 and C_3 in Eq.(12).

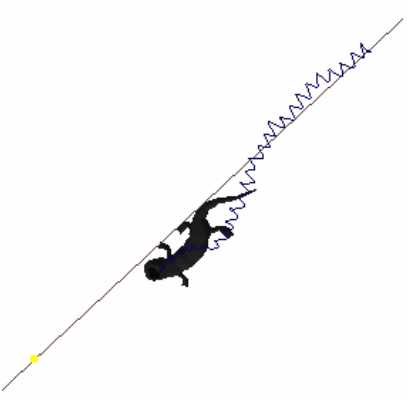
$$f(D,E,S) = -C_1 D - C_2 E + C_3 S \tag{12}$$

6 Experimental Result and Discussion

Simulation experiments are performed in the following settings.



(a) Achieved motion on the ground. The salamander moves by swinging legs alternatively back and forth.



(b) Achieved motion in the water. The salamander moves by swinging its tail.

Fig. 6. Traces of the motions of the salamander model

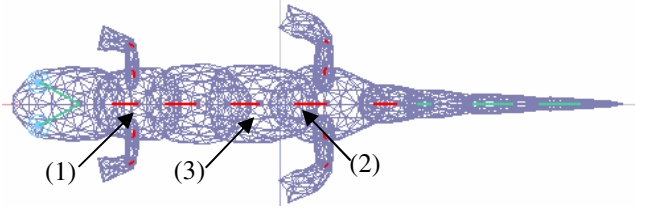


Fig. 7. Recorded position

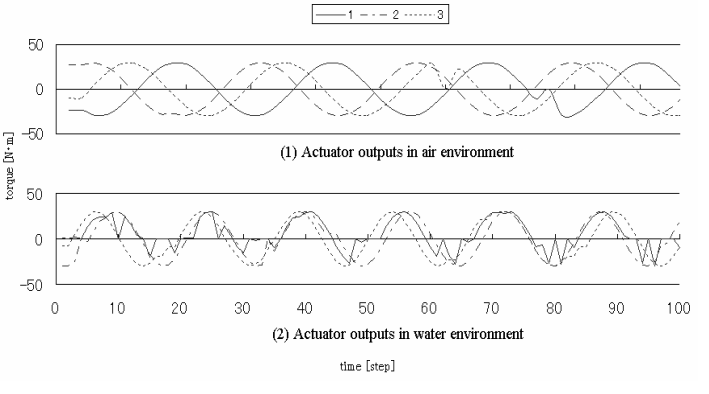


Fig. 8. Actuator outputs of the salamander

1. A salamander is set on the ground. Also, the friction and air influence are implemented with the environment.
2. A salamander is set in the water environment.

Fig.6 shows traces of motions after the salamander moved on the ground or in the water after learning. The salamander acquires effective motions to reach the goal as fast as possible. These motions look like a "walk" behavior on the ground (see Fig.6(a)) and like a "swim" behavior in the water (see Fig.6(b)).

Fig.8 shows some actuator outputs of the salamander which were recorded at positions as shown in Fig.7. When the salamander moves like "walking" on the ground, actuators generate an opposite phases each other between point(1) and point(2). In contrast, actuators generate similar phases in the water. Consequently, "swimming" like behavior is realized by rapid vibration propagation from a head to a tail.

7 Conclusion and Future Work

We aim at establishing a new computer aided animation method. For this purpose, the agent-based and PM-based animation method is introduced. Furthermore, we take consideration in a complex environment, which is dominated by the Newtonian dynamics and fluid influences. Under these circumstances, we treat problems that amphibian behaviors can be acquired adaptively and make its behaviors the animation automatically. Numerical experiments proved that the amphibian put on the ground acquires "walking" and put in the water "swimming". These behaviors are similar to realistic existing organ's behaviors, such as a salamander one in nature. These behaviors are instantly presented in animation. In future work, we would like to introduce multi-agent based animation, where the agents collision, competition, and cooperation are considered.

References

1. Monaghan, J.J.: Smoothed Particle Hydrodynamics. Annual Review of Astronomy and Astrophysics (1992)
2. Koshizuka, S., Nobe, A., Oka, Y.: Numerical Analysis of Breaking Waves Using the Moving Particle Semi-Implicit Method. Numerical Methods in Fluids 26(7), 751–769 (1998)
3. ASSIST, http://rationale.csail.mit.edu/project_assist.shtml
4. Phun, http://www.phun.at/
5. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial Fishes: Autonomous Locomotion, Perception, Behavior, and Learning. Artificial Life 1(4), 327–351 (1994)
6. Ijspeert, A.J., Kodjabachian, J.: Evolution and Development of a Central Pattern Generator for the Swimming of a Lamprey. Artificial Life 5(3), 247–269 (1999)
7. Masry, M., Lipson, H.: A Sketch-Based Interface for Iterative Design and Analysis of 3D Objects. In: Proceedings of Eurographics Workshop on Sketch-Based Interfaces, Dublin, Ireland, pp. 109–118 (2005)
8. Patel, L.N., Murray, A., Hallam, J.: Increased Swimming Control with Evolved Lamprey CPG Controllers. In: International Joint Conference on Neural Networks, IJCNN 2005, vol. 4, pp. 2195–2200 (2005)
9. NVIDIA PhysX, http://www.nvidia.com/object/nvidia_physx.html
10. Hokkaido University, Laboratory of Autonomous System Engineering, Web site, http://junji.complex.eng.hokudai.ac.jp/researches/physics-modeling/movies

Neuro-Evolution Methods for Gathering and Collective Construction

Geoff Nitschke

Department of Computer Science, University of Pretoria, South Africa
gnitschke@cs.up.co.za

Abstract. This paper evaluates the *Collective Neuro-Evolution* (CONE) method, comparative to a related controller design method, in a simulated multi-robot system. CONE solves collective behavior tasks, and increases task performance via facilitating behavioral specialization. Emergent specialization is guided by genotype and behavioral specialization difference metrics that regulate genotype recombination. CONE is comparatively evaluated with a similar *Neuro-Evolution* (NE) method in a *Gathering and Collective Construction* (GACC) task. This task requires a multi-robot system to gather objects of various types and then cooperatively build a structure from the gathered objects. This collective behavior task requires that robots adopt complementary and specialized behaviors in order to solve. Results indicate that CONE is appropriate for evolving collective behaviors for the GACC task, given that this task requires behavioral specialization.

1 Introduction

In fields of research such as *multi-robot systems* [11], it is desirable to reproduce the underlying mechanisms that result in replicating the success of biological collective behavior systems. One such mechanism is *emergent behavioral specialization* [10]. In the study of controller design methods that solve various collective behavior tasks emergent specialization is not used as a problem solving mechanism, but rather emerges as an ancillary result of the system accomplishing its given task. Collective behavior tasks are those requiring cooperative behavior.

This paper applies and tests the *Collective Neuro-Evolution* (CONE) method. CONE is a novel controller design method that solves collective behavior tasks via purposefully facilitating emergent behavioral specialization is currently lacking. CONE adapts a set of *Artificial Neural Network* (ANN) controllers for the purpose of solving collective behavior tasks. The advantage of CONE is that it increases collective behavior task performance or attains collective behavior solutions that could not otherwise be attained without specialization.

In line with state of the art methods for controller design [6], this research supports NE as an appropriate approach for controller design within continuous and partially observable collective behavior task environments. NE has been successfully applied to solve a disparate range of collective behavior tasks that

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 115–123, 2011.

include multi-agent computer games [2], pursuit-evasion games [12], and coordinated movement [1]. Such collective behavior tasks require different agents (controllers) to adopt complementary specialized behaviors in order to solve.

The *Gathering and Collective Construction* (GACC) case study presented in this paper is an initial step towards elucidating that specialization that emerges during controller evolution, can be used as part of a collective behavior problem solving process. Experiments elucidate that CONE, comparative to Multi-Agent ESP, is able to effectuate behavioral specialization in a set of ANN controllers, where such specialization increases collective behavior task performance.

Research Goal: To demonstrate that CONE is appropriate for deriving behavioral specialization in a team of simulated robots, where such specialization gives rise to successful collective construction behaviors.

Hypothesis: CONE facilitates emergent behavioral specialization when evolving a team's collective behavior (in tasks that require specialization), where such specialization contributes to a higher task performance, comparative to the task performance of Multi-Agent ESP evolved teams.

GACC Task: This task requires that a team of simulated robots search an environment and gather a set of atomic objects and then use these objects in the cooperative construction of a complex object. Task performance is measured as the number of atomic objects delivered to a *home area* in a given sequence.

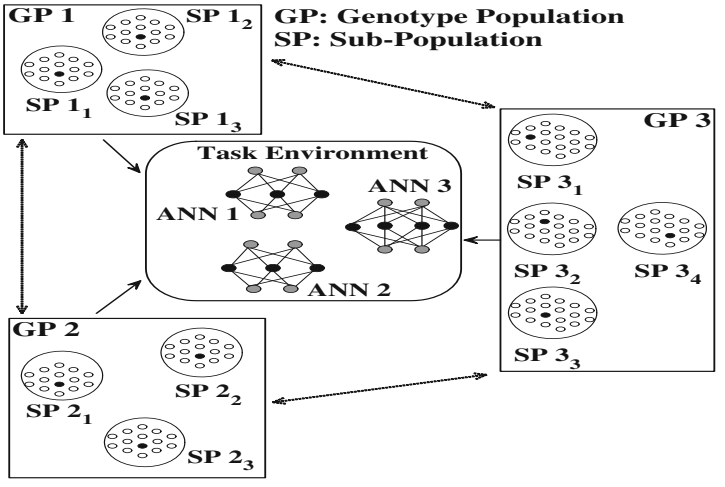


Fig. 1. CONE / Multi-Agent ESP Example. Three ANN controllers are derived from three populations and evaluated in a collective behavior task. CONE: Double ended arrows indicate self regulating recombination occurring *between* populations. Multi-Agent ESP: Recombination only occurs *within* sub-populations.

2 Neuro-Evolution Methods

2.1 Multi-Agent ESP: Multi-Agent Enforced Sub-populations

Multi-Agent ESP is the application of the ESP NE method [8] to collective behavior tasks. Multi-Agent ESP creates n populations for deriving n ANN controllers. Each population consists of u sub-populations, where individual ANNs are constructed as in ESP. This process is repeated n times for n ANNs, which are then collectively evaluated in a task environment. Figure 1 illustrates an example of Multi-Agent ESP using three populations (controllers). Multi-Agent ESP is comprehensively described in related work [12].

2.2 CONE: Collective Neuro-Evolution

Due to space constraints, this section only presents an overview of the novel contributions of *Collective Neuro-Evolution* (CONE). For a comprehensive description of CONE, refer to related work [10]. CONE is a cooperative co-evolution method that adapts a group of ANN controllers. Given n genotype populations, CONE evolves one controller from each population, where controllers must cooperate to solve collective behavior tasks. Controllers are collectively evaluated in a task environment according to how well they solve the given collective behavior task. Each controller is a feed-forward ANN with one hidden layer that is fully connected to the input and output layers. Each hidden layer neuron of each controller is encoded as one genotype. CONE evolves the connection weights of hidden layer neurons, and then combines these neurons into complete controllers. An example of CONE using three controllers (and thus three genotype populations) is presented in figure 1. Unlike related methods such as Multi-Agent ESP, CONE uses *genotype* and *behavioral specialization* (GDM and SDM, respectively) difference metrics to regulate genotype *between* and *within* populations. CONE is an extension of Multi-Agent ESP that includes the following.

1. **GDM:** Adaptively regulates genotype recombination *between* populations, based on neuron (genotype) connection weight similarities [10].
2. **SDM:** Adaptively regulates recombination based on behavioral specialization (CONE uses a specialization metric described in related work [7]) similarities exhibited by controllers [10].
3. **Controller size adaptation:** Adapting the number of hidden layer neurons in each controller facilitates of the evolution of behavioral specialization by CONE, via allowing different controllers to evolve to different sizes. That is, controllers of varying sizes and complexity are often appropriate for solving a set of sub-tasks of varying complexities [10].

2.3 Common Methods

The following describes the procedure used to construct controllers, and evaluate, recombine and mutate genotypes in Multi-Agent ESP and CONE.

- **Constructing Controllers:** Both CONE and Multi-Agent ESP initialize n populations. Population P_i ($i \in \{1, \ldots, n\}$ contains u_i sub-populations. Each sub-population (P_{ij}) contains m genotypes. P_{ij} contains genotypes encoding neurons (strings of floating point values) assigned to position j in the hidden layer of ANN_i. ANN_i is derived from P_i, where $j \leq u_i$.
- **Evaluate all Genotypes:** Systematically select each genotype g in each sub-population of each population, and evaluate g in the context of a complete controller. This controller (containing g) is evaluated together with n-1 other controllers. Other controllers are constructed via randomly selecting a neuron from each sub-population of each of the other populations. The evaluation results in a fitness being assigned to g.
- **Multi-Agent ESP Recombination:** After all neurons (genotypes) have been assigned a fitness [12], each neuron in the *elite portion* (table 1) is recombined with another neuron (randomly selected from the elite portion). Offspring genotypes completely replace each sub-population.
- **CONE Recombination:** The SDM is applied to each pair of controllers. For every pair of controllers within a *Specialization Distance* (SD in table 1) the populations from which these controllers were derived become candidates for recombination. The GDM is then applied to each pair of sub-populations *between* each pair of behaviorally similar populations (controllers). For each pair of sub-populations within a given *Genetic Distance* (GD in table 1) the elite portions of the sub-populations are recombined (using one-point crossover [3]). For every population that is not within the *SD* of another, or each sub-population that is not within the *GD* of another (in another population within the SD),recombination occurs within each sub-population.
- **Mutation:** After recombination, *burst mutation* with a *Cauchy* distribution [8] is applied to each genotype's gene with a given probability (table 1).

3 GACC Task: Experimental Design

Experiments test 30 robots with N building blocks (n type A, p type B, and q type C atomic objects) and a *home area* in a bounded two dimensional continuous environment. The home area (located at the environment's center) is where gathered objects are delivered and where the *complex object* is constructed. A complex object is the structure to be built from N atomic objects. The complex object can only be constructed if robots cooperate place objects of a given type in a predefined sequence. Two, three, and four robots are required to use type A, B, and C objects to construct the complex object. Experiments measure the impact of the Multi-Agent ESP or CONE method and an *environment* upon the number of *atomic objects delivered* in the correct sequence to the home area by the team. The experimental objective test the task performance and the contribution of specialization to performance in teams evolved by each method.

Team Fitness Evaluation: Team fitness (G) equals the total *number of atomic objects delivered in the correct sequence* to the home area. Individual fitness (g_v) equals the *number of atomic objects delivered in the correct sequence* by robot η

Table 1. GACC Simulation and Neuro-Evolution Parameters

Simulation and Neuro-Evolution Parameters	
Number of robots / Genotype populations	30
Robot movement range / cost	0.001 / 0.01
Object/Robot/Home area detection sensor range	0.05
Object/Robot/Home area detection sensor accuracy	1.0
Object/Robot/Home area detection sensor cost	0.01
Robot initial energy	1000 units
Initial robot positions	Random (Excluding home area)
Environment width / height	1.0
Total number of type A, B, and C objects	Variable (table 2)
Atomic Object distribution (Initial positions)	Random (Excluding home area)
Generations / Epochs	250 / 10
Iterations per epoch (Robot team lifetime)	3000
Mutation (per gene) probability / Mutation range	0.05 / [-1.0, +1.0]
Genotype / Specialization distance (CONE)	[0.0, 1.0]
Population elite portion	50%
Weight (gene) range	[-10.0, +10.0]
Genotype length (Number of connection weights)	42
Genotypes per population	500

over the course of its lifetime. The goal of the team is to maximize G. Robots do not maximize G directly, instead each robot η attempts to maximize its own private fitness function g_η, where g_η guides controller evolution.

Simulation: Table 1 presents the simulation and NE parameter settings. These parameter values were determined experimentally. Minor changes to these values produced similar results for both Multi-Agent ESP and CONE. Each experiment consists of 250 generations. One generation is a robot team's *lifetime*. Each robot lifetime lasts for 10 epochs. One epoch is 3000 simulation iterations, and represents a task scenario that tests different robot starting positions, and object locations in an environment. Team task performance is calculated as an average taken over all epochs of a team's lifetime. The best task performance is then selected for each run, and an average is calculated over 20 runs.

4 Robot Sensors, Actuators, and Controller

Detection Sensors: Each robot has eight *object* ([S-0, S-7]), eight *robot* ([S-8, S-15]), eight *home area* ([S-16, S-23]) detection sensors, and three *object demand* ([S-24, S-26]) sensors (figure 2). Each of the eight detection sensors covers one quadrant in a robot's 360 degree sensory Field Of View (FOV). Table 1 presents sensor range, accuracy, and cost.

- **Object Detection Sensors:** Object detection sensors need to be explicitly activated with one of three settings (A, B, C), for detecting type A, B, and C objects, respectively. This constitutes one action and consumes one simulation iteration. Sensor q returns the closest object type (for the current sensor setting) in quadrant q, divided by the squared distance to the robot.
- **Robot Detection Sensors:** The function of these constantly active sensors is to prevent collisions, and provide each robot with an indication of the

current *state* of other robots within *this* robot's sensory FOV. *State* refers to if another robot is carrying an object and the *type* of the object being carried. Sensor q returns a value equal to the object type carried by the closest robot (A:1, B:2, C:3), divided by the squared distance to *this* robot.
- **Home Area Detection Sensors:** Sensor q returns a value inversely proportional to the distance to the home area, divided by the squared distance to *this* robot. Home area sensors are constantly active.
- **Object Demand Sensors:** These constantly active sensors indicate the current demand for object types A, B, C. During the construction process, the complex object broadcasts a signal that is received by each robot's object demand sensors, indicating the next required object type.

Movement Actuators: Two wheel motors control each robot's heading at a constant speed. Movement actuators need to be explicitly activated (motor outputs MO-4 and MO-5 in figure 2). This is one action which takes one simulation iteration. A robot's heading is determined by normalizing and scaling motor output values (vectors dx and dy) by the maximum distance a robot can traverse in one iteration (d_{max}). That is: $dx = d_{max}(o_1 - 0.5)$, and $dy = d_{max}(o_2 - 0.5)$. Where: o_1 and o_2 are values of motor outputs MO-4 and MO-5, respectively.

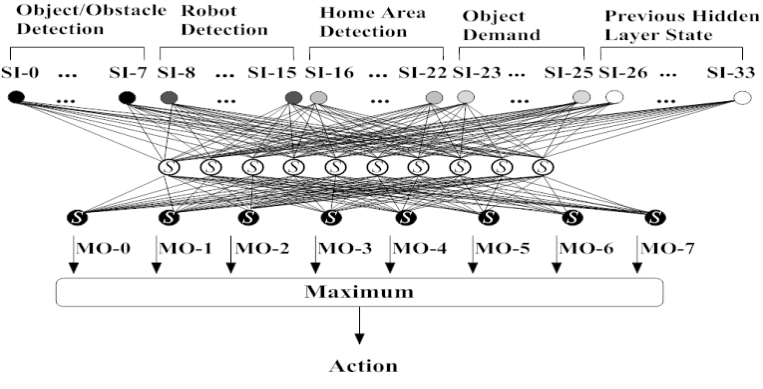


Fig. 2. Robot ANN Controller. For clarity, not all sensory input neurons are illustrated.

Object Gripper: Each robot is equipped with a gripper turret for gripping and transporting objects to the home area. The gripper has three actuator settings (A, B, C) for gripping and transporting type A, B, and C objects, respectively. The gripper needs to be explicitly activated which takes one iteration.

ANN Controller: Each robot uses a recurrent ANN controller [4], which fully connects 34 sensory input neurons to 10 hidden layer neurons to eight motor output neurons (figure 2). Hidden and output neurons are sigmoidal [9] units. Sensory input neurons [SI-26, SI-33] accept input as the previous activation state

of the hidden layer. At each iteration, one of seven actions is executed by a robot. The motor output with the highest value is the action executed.

1. **MO-0:** Activate all object/obstacle detection sensors with setting A.
2. **MO-1:** Activate all object/obstacle detection sensors with setting B.
3. **MO-2:** Activate all object/obstacle detection sensors with setting C.
4. **MO-3, MO-4:** Calculate direction from motor outputs *dx*, *dy*.
5. **MO-5:** Activate gripper with setting A (figure 2).
6. **MO-6:** Activate gripper with setting B (figure 2).
7. **MO-7:** Activate gripper with setting C (figure 2).

5 Results and Discussion

Two experiment sets were run. In experiment set 1, robot teams were evolved in nine *simple environments*. Each simple environment contained a distribution only type A objects, and there was no predefined sequence for object delivery to the home area. In experiment set 2, robot teams were evolved in nine*complex environments*. Each complex environment contained a distribution of type A, B, and C objects. Table 2 presents the distribution of each object type for each environment, and the required sequence that object types must be delivered to the home area in order for the complex object to be constructed. These distributions were derived according to the supposition that if an environment contains multiple object types, there will be a requirement for controllers to specialize to different behaviors in order to efficiently accomplish the task.

Simple Environment Experiment Set: Both Multi-Agent ESP and CONE evolved teams that yielded comparable performance for all simple environments. This result was supported by an independent t-test [5] (P values are not presented due to space constraints). This indicates that environments containing only one object type are not appropriate for encouraging the evolution of behavioral specialization, and that an optimal team behavior in this experiment set is for all controllers to adopt a non-specialized behavior. That is, in the simple environment experiment set, there is no requirement for different controllers to converge to complementary behavioral specializations in order to accomplish

Table 2. Distribution of objects for each complex environment. Env: Environment.

Env	Object-A Number	Object-B Number	Object-C Number	Complex Object Build Sequence
1	1	2	7	CCACBCBCCC
2	2	4	4	CAACBBBCBC
3	3	6	1	BABAABCBBB
4	16	2	2	BAAAABAAAACAAAAAAAAC
5	4	14	2	BBBABABBBBABABBBCBCB
6	7	2	11	BACACACACACACACCCCCB
7	12	13	5	CBAABCAACAABBBAACABABBBAACBBBB
8	13	14	3	BABABABAAACABABAABBBBCAACABBBB
9	4	15	11	CBBBCBCACBBBCACCACBBBCACBBBBBC

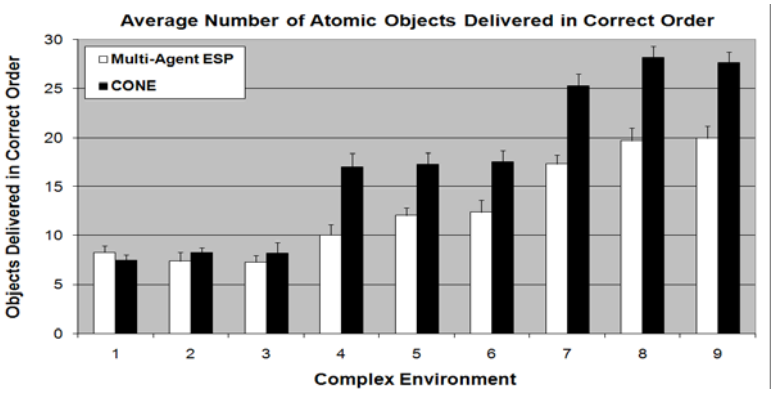


Fig. 3. Average Number of Objects Delivered in Correct Order to Home Area in each Complex Environment by Multi-Agent ESP and CONE evolved teams

the task. This result supports the hypothesis that CONE only evolves and uses behavioral specialization in tasks that require specialization.

Complex Environment Experiment Set: Figure 3 presents, for each complex environment, the average *number of atomic objects* delivered to the construction zone in the correct order, by Multi-Agent ESP and CONE evolved teams. Task performance results presented in figure 3, indicates that CONE evolved teams yield a significantly higher average task performance comparative to Multi-Agent ESP evolved teams in six out of the nine environments. This is supported by an independent t-test. In each complex environment, the fittest CONE evolved team consisted of multiple castes (robot sub-groups specialized to gathering and construction with either type A, B, or C objects). This result supports the hypothesis that CONE is appropriate for deriving behavioral specialization which leads to a higher collective behavior task performance (comparative to Multi-Agent ESP evolved teams) is achieved.

6 Conclusions

This paper described the application of the Multi-Agent ESP and CONE neuroevolution methods for the purpose of automating controller design in a team of simulated robots. CONE effectively facilitated specialization in the behaviors of the robots which (comparative to Multi-Agent ESP teams) lead to a higher task performance in a process in a gathering and collective construction task. This research suggests that the controller design process used by CONE is able to leverage and use emergent specialization in tasks that benefit from a behaviorally specialized problem solving approach.

References

1. Baldassarre, G., Nolfi, S., Parisi, D.: Evolving mobile robots able to display collective behavior. Artificial Life 9(1), 255–267 (2003)
2. Bryant, B.: Evolving Visibly Intelligent Behavior for Embedded Game Agents. PhD thesis. Computer Science Department. University of Texas, Austin, USA (2006)
3. Eiben, A., Smith, J.: Introduction to Evolutionary Computing. Springer, Berlin (2003)
4. Elman, J.: Finding structure in time. Cognitive Science 14(1), 179–211 (1990)
5. Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes. Cambridge University Press, Cambridge (1986)
6. Floreano, D., Dürr, P., Mattiussi, C.: Neuroevolution: from architectures to learning. Evolutionary Intelligence 1(1), 47–62 (2008)
7. Gautrais, J., Theraulaz, G., Deneubourg, J., Anderson, C.: Emergent polyethism as a consequence of increased colony size in insect societies. Journal of Theoretical Biology 215(1), 363–373 (2002)
8. Gomez, F., Miikkulainen, R.: Incremental evolution of complex general behavior. Adaptive Behavior 5(1), 317–342 (1997)
9. Hertz, J., Krogh, A., Palmer, R.: Introduction to the Theory of Neural Computation. Addison-Wesley, Redwood City (1991)
10. Nitschke, G.: Neuro-Evolution for Emergent Specialization in Collective Behavior Systems. PhD thesis. Computer Science Department, Vrije Universiteit., Amsterdam, Netherlands (2009)
11. Schultz, C., Parker, L.: Multi-robot Systems: From Swarms to Intelligent Automata. Kluwer Academic Publishers, Washington DC, USA (2002)
12. Yong, C., Miikkulainen, R.: Coevolution of Role-Based Cooperation in Multi-Agent Systems. Technical Report AI07-338. Department of Computer Sciences. The University of Texas, Austin, USA (2007)

On the Dynamics of Active Categorisation of Different Objects Shape through Tactile Sensors

Elio Tuci, Gianluca Massera, and Stefano Nolfi

ISTC-CNR, Via San Martino della Battaglia, n. 44
00185 Rome, Italy
{elio.tuci,gianluca.massera,stefano.nolfi}@istc.cnr.it

Abstract. Active perception refers to a theoretical approach to the study of perception grounded on the idea that perceiving is a way of acting, rather than a process whereby the brain constructs an internal representation of the world. In this paper, we complement previous studies by illustrating the operational principles of an active categorisation process in which a neuro-controlled anthropomorphic robotic arm, equipped with coarse-grained tactile sensors, is required to perceptually categorise spherical and ellipsoid objects.

Keywords: Active perception, categorisation, evolutionary robotics.

1 Introduction

Categorical perception is a fundamental cognitive capacity displayed by natural organisms, and it can be defined as the ability to divide continuous signals received by sense organs into discrete categories whose members resemble each other more than members of other categories [1]. Most of the work in literature focuses on categorization processes that are passive (i.e., the agents can not influence their sensory states through their actions) and instantaneous (i.e., the agents are demanded to categorise their current sensory state rather than a sequence of sensory states distributed over a certain time period). Active categorical perception can be studied by exploiting the properties of autonomous embodied and situated agents, in which categorical perception is strongly influenced by the agent action [see also 2, 3, on this issue].

The works described in [4, 5, 6, 7, 8, 9, 10] contributed to the study of active categorisation by showing that relatively complex categorisation tasks can be solved by autonomous agents equipped with simple sensory-motor and cognitive apparatus that lacks some of those elements previously assumed necessary to recognise and categorise various types of objects or environmental circumstances. By following this line of investigation, the work described in [11] focuses on the study of categorical perception in a task in which a simulated anthropomorphic robotic arm is demanded to actively categorize un-anchored spherical and ellipsoid objects placed in different positions and orientations over a planar surface. Populations of evolving robots are left free to determine the way in which they categorize the shape of the objects within the limits imposed by the

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 124–131, 2011.

experimental scenario and by the computational power of their neural controller. This implies that the robots are left free to determine (i) how to interact with the external environment (by eventually modifying the environment itself); (ii) how the experienced sensory stimuli are used to discriminate the two categories; and (iii) how to represent in the categorisation space each object category. The analysis of the obtained results indicates that the robots are indeed capable of developing an ability to effectively categorize the shape of the objects despite the high similarities between the two types of objects, the difficulty of effectively controlling a body with many degrees of freedoms (hereafter, DOFs), and the need to master the effects produced by gravity, inertia, collisions etc. More specifically, the best individuals display an optimal ability to correctly categorize the objects located in different positions and orientations already experienced during evolution, as well as an excellent ability to generalize their skill to objects positions and orientations never experienced during evolution.

This paper complements the results and analysis shown in [11] by describing interesting operational aspects of the categorisation ability of the best evolved agent. In particular, we look at (i) how the robot acts in order to bring fourth the sensory stimuli which provide the regularities necessary for categorizing the objects in spite of the fact that sensation itself may be extremely ambiguous, incomplete, partial, and noisy; (ii) the dynamical nature of sensory flow (i.e., how sensory stimulation varies over time and the time rate at which significant variations occur); (iii) the dynamical nature of the categorization process (i.e., whether the categorization process occur over time while the robot interacts with the environment).

2 Methods

In this Section, we provide only a minimal description of the methods employed to design successful controllers. More details on the methods of this study can be found in [11]. The simulated robot consists of an anthropomorphic robotic arm with 7 actuated DOFs and a hand with 20 actuated DOFs (see Fig. 1a). Proprioceptive and tactile sensors are distributed on the arm and the hand (see Fig. 1b and 1c). The robot and the robot/environmental interactions are simulated using Newton Game Dynamics (NGD), a library for accurately simulating rigid body dynamics and collisions (more details at `www.newtondynamics.com`). The active joints of the robotic arm are actuated by two simulated antagonist muscles implemented accordingly to the Hill's muscle model, as detailed in [12]. The agent controller consists of a continuous time recurrent non-linear network (CTRNN) with 22 sensory neurons, 8 internal neurons, and 18 motor neurons [see Fig. 1d and also 13]. $\tau_i \dot{y_i} = -y_i + gI_i$ for $i = 1, .., 22$ is the equation used to update the state of sensory neurons. $\tau_i \dot{y_i} = -y_i + \sum_{j=n}^{m} \omega_{ji} \sigma(y_j + \beta_j)$ for $i = 23, .., 30$; with $n = 1$, and $m = 30$ is the equation used to update the state of internal neurons, and for $i = 31, .., 48$; with $n = 23$, and $m = 30$ is used to update the state of motor neurons. y_i represents the state of a neuron, τ_i the decay constant, g is a gain factor, I_i the intensity of the perturbation on sensory neuron i, ω_{ji} the

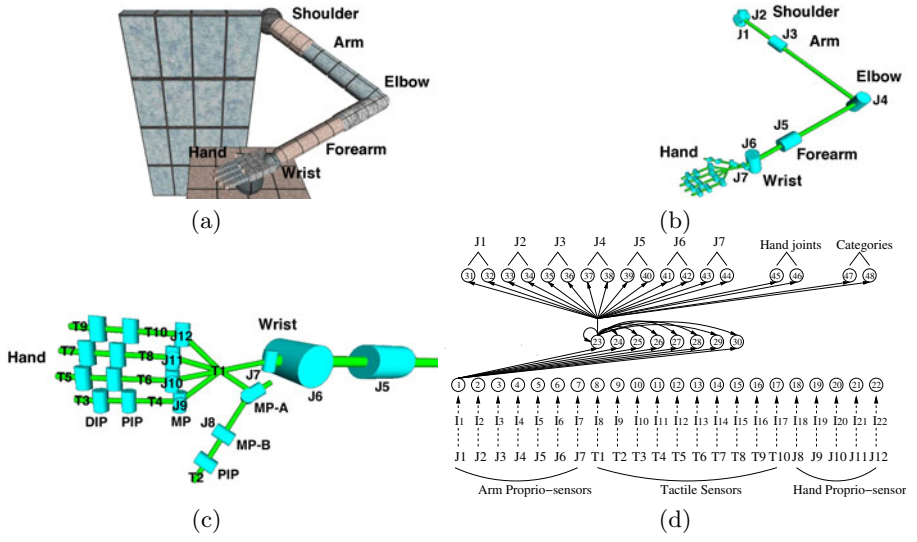


Fig. 1. (a) The simulated robotic arm. The kinematic chain (b) of the arm, and (c) of the hand. In (b) and (c), cylinders represent rotational DOFs; the axes of cylinders indicate the corresponding axis of rotation; the links among cylinders represents the rigid connections that make up the arm structure. J_i with $i = 1, .., 12$ refer to the joints whose state is both sensed and set by the arm's controller. T_i with $i = 1, .., 10$ indicate the tactile sensors. (d) The architecture of the arm controller. The circles refer to the artificial neurons. Continuous line arrows indicate the efferent connections for the first neuron of each layer. Dashed line arrows indicate the correspondences between joints and tactile sensors and input neurons. The labels on the dashed line arrows refer to the mathematical notation used to indicate the readings of the corresponding sensors.

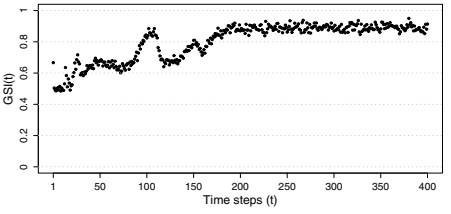
strength of the synaptic connection from neuron j to neuron i, β_j the bias term, $\sigma(y_j+\beta_j)$ the firing rate. τ_i with $i = 23, .., 30$, β_i with $i = 1, .., 48$, all the network connection weights ω_{ij}, and g are genetically specified networks' parameters. τ_i with $i = 1, .., 22$ and $i = 31, .., 48$ is equal to ΔT. There is one single bias for all the sensory neurons. The activation values y_i of motor neurons determine the state of the simulated muscles of the arm [see 12, for a detailed description of the functional properties of the arm]. The activation values y_i of output neurons $i = 47, 48$ are used to categorize the shape of the objects. In particular, in each trial k, the agent represents the experienced object (i.e., the sphere S or the ellipsoid E) by associating to it a rectangle R_k^S or R_k^E whose vertices are: $(\min_{0.95T<t<T} \sigma(y_{47}(t) + \beta_{47}), \min_{0.95T<t<T} \sigma(y_{48}(t) + \beta_{48}))$ for the bottom left vertex, and $(\max_{0.95T<t<T} \sigma(y_{47}(t) + \beta_{47}), \max_{0.95T<t<T} \sigma(y_{48}(t) + \beta_{48}))$ for the top right vertex, with $T = 400$ time steps (i.e., 4 simulated seconds) corresponding to the length of a trial. The sphere category, referred to as C^S, corresponds to the minimum bounding box of all R_k^S; the ellipsoid category, referred to as C^E, corresponds to the minimum bounding box of all R_k^E.

A simple generational genetic algorithm is employed to set the parameters of the networks [see 14]. The initial population contains 100 genotypes. Generations following the first one are produced by a combination of selection with elitism, and mutation [see also 11, for details]. Cell potentials are set to 0 when the network is initialised or reset (i.e., at the beginning of each trial), and circuits are integrated using the forward Euler method with an integration step-size $\Delta T = 0.01$ [see 15]. During evolution, agents have been rewarded by an evaluation function which seeks to assess their ability to recognise and distinguish the ellipsoid from the sphere. Note that, rather than imposing a representation scheme in which different categories are associated with *a priori* determined state/s of the categorization neurons (i.e., neurons 47 and 48), we left the robots free to determine how to communicate the result of their decision. That is, the agents can develop whatever representation scheme as long as each object category is clearly identified by a unique state/s of the categorisation neurons. More precisely, we scored agents on the basis of the extent to which the categorization outputs produced for objects of different categories are located in non-overlapping regions of a two dimensional categorization space $C \in [0,1] \times [0,1]$.

3 Results

Results of post-evaluation tests illustrated in [11] shows that the best evolved agent (hereafter, A_1) possesses a close to optimal ability to discriminate the shape of the objects as well as an excellent ability to generalize their skill in new circumstances. Moreover, in [11] it is shown that A_1, for one of the two positions experienced during evolution (i.e., position A, angle of joints $J_1, ..., J_7$ are $\{-50°, -20°, -20°, -100°, -30°, 0°, -10°\}$), exploits only tactile sensation to categorise the objects. In this Section, we take advantage of this latest result by running tests that further explore the dynamics of the decision of A_1 in position A, beyond the qualitative description illustrated in [11]. In particular, our interest is in finding out whether there are distinctive and functionally different temporal phases during the categorisation process. How long does the agent need to interact with the object before been able to tell whether is touching a sphere or an ellipsoid? Does the discrimination process occur at a specific moment, as a response to a sensory pattern that encode the regularities which are necessary for discriminating, or does it occur over time by integrating the information contained in several successive sensory states? Note that movies of the best evolved strategies can be found at `http://laral.istc.cnr.it/esm/active_perception`.

To answer these questions we begin by using a slightly modified version of the Geometric Separability Index (hereafter, referred to as *GSI*) originally proposed in [16]. GSI represents an estimate of the degree to which tactile sensor readings experienced during the interactions with the sphere or with the ellipsoid are separated in sensory space. We built four hundred data sets, one for each time step with the ellipsoid (i.e., $\{\tilde{I}_k^E\}_{k=1}^{180}$), and four hundred data sets, one for each time step with the sphere (i.e., $\{\tilde{I}_k^S\}_{k=1}^{180}$). Where, $\tilde{I}_k^E$ is the tactile sensor readings



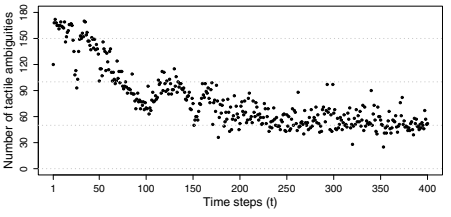
(a)

$$GSI(t) = \frac{1}{K}\sum_{k=1}^{K} z_i, \text{ with } K = 180;$$

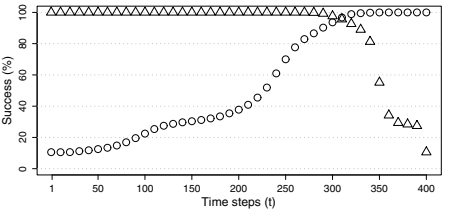
$$z_k = \begin{cases} 1 & if m^{EE} < m^{ES}; \\ 0 & if m^{EE} > m^{ES}; \\ \frac{u}{u+v} & otherwise; \end{cases}$$

$$m^{EE} = min(H(\tilde{I}_k^E, \tilde{I}_j^E))$$
$$m^{ES} = min(H(\tilde{I}_k^E, \tilde{I}_j^S))$$
$$u = |\{\tilde{I}_j^E : H(\tilde{I}_k^E, \tilde{I}_j^E) = m^{EE}\}_{j=1}^{K}|$$
$$v = |\{\tilde{I}_j^S : H(\tilde{I}_k^E, \tilde{I}_j^S) = m^{ES}\}_{j=1}^{K}|$$

(b)



(c)



(d)

Fig. 2. (a) The Geometric Separability Index (GSI); (b) the formal definition of GSI; (c) the number of tactile ambiguities; (d) the percentage of success in *pre-substitution tests* (see triangles) and *post-substitution tests* (see empty circles)

experienced by A_1 while interacting with the ellipsoid at time step t of trial k; and $\tilde{I}_k^S$ is the tactile sensor readings experienced by A_1 while interacting with the sphere at time step t of trial k. Trial after trial, the initial rotation of the ellipsoid around the z-axis changes of 1°, from 0° in the first trial to 179° in the last trial. Each trial is differently seeded to guaranteed random variations in the noise added to sensors readings. At each time step t, the *GSI* is computed as shown in Fig. 2b, where H(x, y) is the Hamming distance between tactile sensor readings. $|x|$ means the cardinality of the set x. *GSI*=1 means that at time step t the closest neighbourhood of each $\tilde{I}_k^E$ is one or more $\tilde{I}_k^E$. *GSI*=0 means that at time step t the closest neighbourhood of each $\tilde{I}_k^E$ is one or more $\tilde{I}_k^S$.

As shown in Fig. 2a, the *GSI(t)* tends to increase from about 0.5 at time step 1 to about 0.9 at time step 200, and to remain around 0.9 until time step 400. This trend suggests that during the first 200 time steps, the agent acts in a way to bring forth those tactile sensor readings which facilitate the object identification and classification task. In other words, the behaviour exhibited by the agent allows it to experience two classes of sensory states, rather well separated in the sensory space, which correspond to objects belonging to two different categories. However, the fact that the *GSI* does not reach the value of 1.0 indicates that the two groups of sensory patterns belonging to the two

objects are not fully separated in the sensory space. In other words, some of the sensory patterns experienced during the interactions with an ellipsoid are very similar or identical to sensory patterns experienced during interactions with the sphere and vice versa. This is confirmed by the graph shown in Fig. 2c, which refers to the number of tactile ambiguities at each time step. A tactile ambiguity is defined as the condition in which $m^{ES} = 0$. This means that some of the patterns are experienced during interactions with both an ellipsoid and a sphere. This implies that A_1 can not determine the category of the current object solely on the basis of the current sensory stimuli. Thus, it follows that the most plausible hypothesis about the categorization process is that it involves an ability to integrate sequences of experienced sensory states over time. To test this hypothesis we employ *substitution tests*.

A *substitution test* is a post-evaluation test in which one type of sensory information experienced by the agent during the interactions with an ellipsoid is replaced with the corresponding type of sensory information previously recorded in trials in which the agent was interacting with a sphere. In this case, we replace tactile sensation at specific interval of time during each trial. In a first series of tests, referred to as *pre-substitution tests*, substitutions have been applied from the beginning of each trial up to time step t where t = 1,..,400. In a second series of tests, referred to as *post-substitution tests*, substitutions have been applied from time step t, where t = 1,..,400, to the end of a trial t=400. Each test has been repeated at intervals of 20 time steps. The test is repeated for 180 trials in which the orientation of the ellipsoid object around the z-axis varies from 0°, in the first trial, to 179°, in the last trial. In a *substitution test*, a 400 time steps trial k can: (i) successfully terminate if the R_k^E, built as illustrated in Sec. 2, completely falls within the bounding box C^E, previously built by running specific post-evaluation tests, and corresponding to the ellipsoid category for agent A_1; (ii) unsuccessfully terminate with a sphere response if the R_k^E completely falls within the two-dimensional space delimited by the bounding box C^S previously built by running specific post-evaluation tests, and corresponding to the sphere category for agent A_1; (iii) unsuccessfully terminate with a none response, if the R_k^E, completely falls outside the two-dimensional space delimited by the bounding boxes $C_i^S \cap C_i^E$. The results of *pre-substitution tests* and *post-substitution tests* are illustrated in Fig. 2d, which shows that, regardless of the rotation of the ellipsoid, pre-substitutions which do not affect the last 100 time steps do not cause any drop in performance. For *pre-substitution tests* that involve more than 300 time steps the amount of performance drop is higher for longer substitution periods (see triangles in Fig. 2d). Similarly, the agent does not incur in any performance drop if post-substitutions affect less than 100 time steps. For *post-substitution tests* that affect more than the last 100 time steps the amount of performance drop is higher for longer substitution periods (see empty circles in Fig. 2d). Overall, the results shown in Fig. 2 as well as the trajectories of the average decision outputs shown in [11] indicate that, for what concerns position A, the interactions between the agent and the objects can be divided into three temporal phases that are qualitatively different from the point of view of the

categorization process: (i) an initial phase whose upper bound can be approximately fixed at time step 250, in which the categorization process begins but in which the categorization answer produced by the agent is still reversible; (ii) an intermediate phase whose upper bound can be approximately fixed at time step 350, in which very often a categorization decision is taken on the basis of all previously experienced evidences; and (iii) a final phase in which the previous decision (which is now irreversible) is maintained. As also noticed by looking at the trajectories of the average decision outputs shown in [11], during the initial phase the robot starts to differentiate the categorization output produced for different type of objects by accumulating the evidences provided by the experienced sensory states. The fact the sensory states provide sufficient information for discriminating the two categories is demonstrated by the fact that the *GSI* increases from the chance level (0.5) up to a value of about 0.9 at the end of the initial phase (see Fig. 2a). The fact that the categorization decision formed by the agent during the initial phase is not definitive yet is demonstrated by the fact that substitutions of the critical sensory stimuli performed during this phase do not cause any drop in performance (see Fig. 2d, triangles). The fact that the intermediate phase corresponds to a critical period is demonstrated by the fact that pre-substitutions and post-substitutions affecting this phase produce a significant drop in performance (see Fig. 2d). The fact that the robot take an ultimate decision during the intermediate phase is demonstrated by the fact that post-substitutions affecting the last 80 time steps, approximately, do not produce any drop in performance (see Fig. 2d, empty circles).

4 Conclusion

This paper illustrates post-evaluation tests that complement the results shown in [11] concerning the perceptual categorisation ability of a simulated autonomous agent. The analysis indicates that one fundamental skill that allows the best evolved agent to distinguish sphere from ellipsoid objects consists in the ability to interact with the external environment and to modify the environment itself so to experience sensory states which are as differentiated as possible for different categorical contexts. On the one hand, this result represents a confirmation of the importance of sensory-motor coordination, and more specifically of the active nature of situated categorization, already highlighted in previous studies [e.g., 7, 8]. On the other hand, the results demonstrate that, in this specific scenario, sensory-motor coordination needs to be complemented by other additional mechanisms. In fact, the best evolved robot does not succeed in acting in a way to experience at any time step separated sensory states for different object categories. The categorization process displayed by this agent is realized dynamically by integrating the evidences provided by the experienced sensory stimuli over time.

Acknowledgments. This research work was supported by the *ITALK* project (EU, ICT, Cognitive Systems and Robotics Integrating Project, grant n° 214668).

References

[1] Cohen, H., Lefebvre, C. (eds.): Handbook of Categorisation in Cognitive Science. Elsevier, Amsterdam (2005)

[2] Gibson, J.J.: The theory of affordances. In: Shaw, R., Bransford, J. (eds.) Perceiving, Acting and Knowing. Toward an Ecological Psychology, pp. 67–82. Lawrencw Erlbaum Associates, Hillsdale (1977)

[3] Noë, A.: Action in Perception. MIT Press, Cambridge (2004)

[4] Nolfi, S.: Power and limits of reactive agents. Neurocomputing 49, 119–145 (2002)

[5] Beer, R.: The dynamics of active categorical perception in an evolved model agent. Adaptive Behavior 11, 209–243 (2003)

[6] Williams, P., Beer, R., Gasser, M.: An embodied dynamical approach to relational categorization. In: Love, B., McRae, K., Sloutsky, V. (eds.) Proc. of the 30th Annual Conference of the Cognitive Science Society, pp. 223–228. Cognitive Science Society, Inc. (2008)

[7] Scheier, C., Pfeifer, R., Kunyioshi, Y.: Embedded neural networks: exploiting constraints. Neural Networks 11(7-8), 1551–1596 (1998)

[8] Nolfi, S., Marocco, D.: Active perception: A sensorimotor account of object categorisation. In: Hallam, B., Floreano, D., Hallam, J., Hayes, G., Meyer, J.A. (eds.) Proc. of the 7th International Conference on Simulation of Adaptive Behavior (SAB 2002), pp. 266–271. MIT Press, Cambridge (2002)

[9] Gigliotta, O., Nolfi, S.: On the coupling between agent internal and agent/ environmental dynamics: Development of spatial representations in evolving autonomous robots. Adaptive Behavior 16, 148–165 (2008)

[10] Tuci, E., Trianni, V., Dorigo, M.: Feeling the flow of time through sensory-motor coordination. Connection Science 16(4), 301–324 (2004)

[11] Tuci, E., Massera, G., Nolfi, S.: Active categorical perception in an evolved anthropomorphic robotic arm. In: Proc. of the IEEE Conference on Evolutionay Computation (CEC 2009), Special Session on Evolutionary Robotics (2009), ISBN: 978-1-4244-2959-2, Draft available at `http://laral.istc.cnr.it/elio.tuci/pagn/pubb.html`

[12] Massera, G., Cangelosi, A., Nolfi, S.: Evolution of prehension ability in an anthropomorphic neurorobotic arm. Front. Neurorobot. 1 (2007)

[13] Beer, R., Gallagher, J.: Evolving dynamic neural networks for adaptive behavior. Adaptive Behavior 1(1), 91–122 (1992)

[14] Goldberg, D.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading (1989)

[15] Strogatz, S.: Nonlinear Dynamics and Chaos. Perseus Books Publishing, Cambridge (2000)

[16] Thornton, C.: Separability is a learner's best friend. In: Bullinaria, J., Glasspool, D., Houghton, G. (eds.) Proc. of the 4th Neural Computation and Psychology Workshop: Connectionist Representations, pp. 40–47. Springer, London (1997)

Evolving a Novel Bio-inspired Controller in Reconfigurable Robots*

Jürgen Stradner, Heiko Hamann, Thomas Schmickl,
Ronald Thenius, and Karl Crailsheim

Artificial Life Laboratory of the Department of Zoology
Karl-Franzens University Graz, Universitätsplatz 2, A-8010 Graz, Austria
{juergen.stradner,heiko.hamann,thomas.schmickl,ronald.thenius,
karl.crailsheim}@uni-graz.at

Abstract. Evolutionary robotics uses evolutionary computation to optimize physically embodied agents. We present here a framework for performing off-line evolution of a pluripotent robot controller that manages to form multicellular robotic organisms from a swarm of autonomously moving small robot modules. We describe our evolutionary framework, show first results and discuss the advantages and disadvantages of our off-line evolution approach. In detail, we explain the single parts of the framework and a novel homeostatic hormone-based controller, which is shaped by artificial evolution to control both, the non-aggregated single robotic modules and the joined high-level robotic organisms. As a first step we present results of this evolutionary shaped controller showing the potential for different motion behaviours.

1 Introduction

Recently evolutionary robotics (ER) has become a fascinating field that exploits evolutionary computation (EC) to optimize physically embodied agents. In some studies robot controllers were adapted by EC in simulated worlds [1]. In contrast to that, other studies [2] showed evolution of single robots and small robot groups in a process of on-line evolution, where a sort of genetic algorithm (GA) was optimizing artificial neural networks (ANN) during runtime of the real robot(s). The advantages and disadvantages of both approaches are clear: On-line evolution profits from the fact that real hardware is used in real-world environments but suffers from lower number of generations. Off-line evolution profits from computational speed (parallel processing, grids) but suffers from differences between models and reality [3]. As an intersection of these approaches, the swarm-bots project [4] used a set of simulation tools of varying levels of detail (physics, robot model) to perform off-line evolution of ANNs, which were finally tested on real robotic hardware.

In former studies we used an evolutionary strategy [5] to shape algorithms that aggregated robots autonomously in various group sizes at target areas [6].

* Supported by: EU-IST-FET project 'SYMBRION', no. 216342; EU-ICT project 'REPLICATOR', no. 216240.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 132–139, 2011.

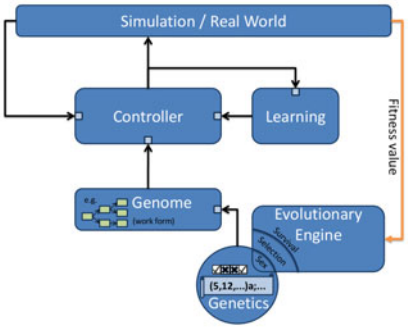


Fig. 1. Schematic overview of the parts of our framework for swarm-level off-line evolution of a robotic swarm. See text for details

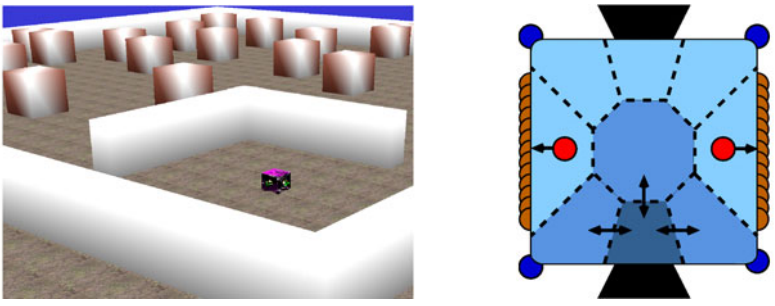
The projects SYMBRION [7] and REPLICATOR [8] have goals far beyond pure swarm robotics: Hundreds of robot modules will autonomously explore the environment, collect and distribute information. In specific situations the swarm of robots will aggregate, the robotic modules will join together to various body shapes and will overcome obstacles and barriers which a single robot module would not be able to overcome.

In the article at hand, we describe our approach to the above mentioned set of challenges by means of off-line evolution. This approach is chosen to achieve real robot behavior (planned future work) as we expect that a high number of generations will be needed using artificial evolution (AE). Our evolutionary framework is structured in several software components, thus, experiments could be performed in several variants with little overhead. The main components of our framework are: simulation environment, runtime controller and its genome, genetic component and evolutionary engine. The interplay between these components is depicted in Fig. 1 and they are described in detail in section 2.

2 Framework for Off-Line Evolution

2.1 Simulation: Symbricator3DSimulator

The simulation environment for our experiments is Symbricator3D (Fig. 2(a)), based on "Delta3D" [9], which is an open source gaming and simulation engine. Our evolutionary framework is implemented within this software package. The simulation is equipped with an embedded physics engine (ODE) which keeps the gap between simulation and real-world as small as possible. For our ambition to transfer the results of the off-line simulation onto real robots the actors in the simulation are close representatives of the robots used in the projects SYMBRION and REPLICATOR. These cubic robots with a side length of approximately 5cm (for a schematic graphic see Fig. 2(b)) are still in development. However, the most important design parameters are already implemented into the simulation. The actuation is performed by two screws, which allow motion in



(a) (b)

Fig. 2. (a) Screenshot of the simulation environment in Symbricator3D. (b) Schematic graphic of the robot. Double arrows symbolise the diffusion between the compartments, arrows with a circle mark the effect of a hormone on the actuators (i.e., the screws that are indicated at the sides of the robot).

all horizontal directions. Their position and orientation is indicated in Fig. 2(b). Hinges establish connections between single modules and allow a 3D formation of the robot organism by changing the angle between the connected modules. It is equipped with 12 distance sensors.

2.2 Controller

One major novelty of our framework is the use of a robot controller that controls the movements of the modules during the swarm phase and that also regulates the body formation process of the robotic organisms. In addition, the very same controller is used to control the body movements of the joined high-level organisms. In nature, the aggregation of the slime mould *Dictyostelium discoideum* shows that this is achieved by one chemical signal ("cAMP") and a fixed set of rules executed by each slime mould amoeba. The robot receives information by its sensors and moves and turns by activating its screws. The basic idea of our artificial homeostatic hormone controller (AHHS, [10]) is that it mimics the endocrine processes that lead to internal homeostasis in real organisms. The virtual hormones are described by their chemical/physical properties: production rate, decay rate and diffusion coefficient. In the genome (see Sec. 2.3), a table of rules describes how sensors affect hormones, how hormones affect actuators, and how hormones interfere with each other. We plan to use this system also within the joined robot organism, by allowing the hormones to diffuse also from one robot to another. The spreading of the hormone through the aggregated organism controls the body-formation process. In addition, the hormones are used to generate a synchronized movement of the organism's body. The AHHS-based controller is used not only in the robotic organism, but also during the swarm phase of the organism. We already developed several AHHS controllers by hand that were tested successfully. These controllers allow the robots to perform basic navigation tasks. In this paper, we present the performance of the controller shaped by AE as a first result of our evolutionary framework.

2.3 Genome

The genome of the AHHS consists of two logical entities: *hormone chromosome* and *rule chromosome*. The hormone chromosome appears once in the genome for each hormone and the rule chromosome appears arbitrarily often for each hormone (depending on how many rules are active/possible per hormone).

The hormone chromosome contains the following parameters:

- *hormone ID*
- *fixed decay rate*
- *diffusion coefficient*
- *maximum value of hormone* (value at which a saturation is reached)
- *base production rate* (amount that is produced per time step without sensory stimulation)

The rule chromosome contains the following parameters:

- *rule type*: condition to be met or triggering action
  1. **always:** Action triggered independent from threshold σ
  2. **greater than:** Action triggered if greater than threshold σ
  3. **smaller than:** Action triggered if smaller than threshold σ
- *trigger type*: type of triggered action (hormone concentration θ, actuator value α, dependent dose δ, fixed dose β, sensor value γ)
  1. **never triggered:** No action performed.
  2. **actuator influences hormone:** if $(\alpha(t) > \sigma)$ then $\theta(t+1) = \theta(t) + \alpha(t)\delta + \beta$
  3. **sensor influences hormone:** if $(\gamma(t) > \sigma)$ then $\theta(t+1) = \theta(t) + \gamma(t)\delta + \beta$
  4. **hormone influences actuator:** if $(\theta(t) > \sigma)$ then $\alpha(t+1) = \alpha(t)\delta + \beta$
  5. **hormone influences other hormone:** if $(\theta_1(t+1) > \sigma)$ then $\theta_2(t+1) = \theta_2(t) + \theta_1(t)\delta + \beta)$
  6. **hormone influences itself:** $\theta(t+1) = \theta(t) + \theta(t)\delta + \beta$

All these values are integer values allowing fast execution of these rules on limited (embedded) hardware.

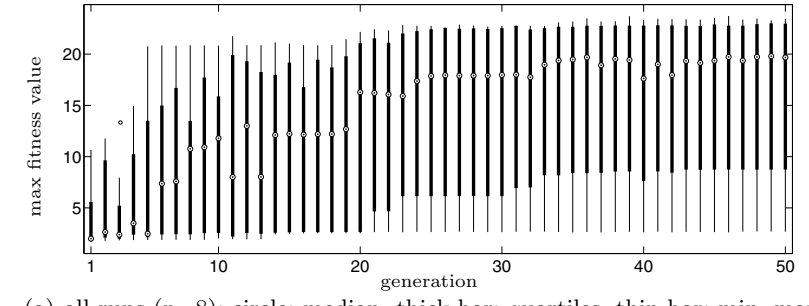
2.4 Evolutionary Engine and Genetics

The evolutionary engine represents the implementation of an evolutionary algorithm (EA) [11] adapted to our swarm and joined robot organism. The evolvable individual consists of one genome setting and its associated robot module(s). The evaluation function is adapted to the specific task (e.g., value of distance sensors, movement). In our experiments the evolving population consists of 20 single robot modules, exploring the environment consecutively. The parent selection is non-linear proportional to the fitness of individuals and the mutation rate set to 0.2 per hormone and rule. Elitism is set to 3.

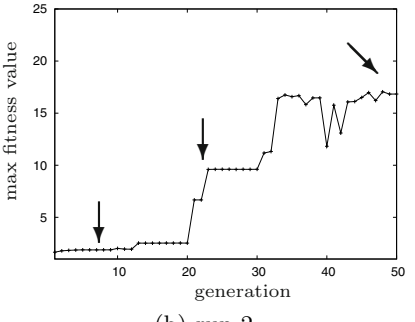
The evolutionary process operates at various places on the configuration of our AHHS-controller: It alters the rule set, the basic properties of sensor input and of actuator output. In addition, it alters the basic virtual chemical/physical properties of hormones. All these modifications significantly change the behavior that is produced by the controller. Due to hormone-to-hormone interactions, various complex behaviors emerge, thus our controller is "pluripotent".

3 Evaluating the AHHS-Controller

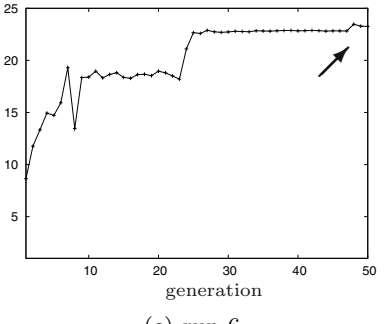
To test the functionality of the evolutionary framework and the evolvability of the AHHS controller a task called "explore the environment" was performed. The controller had to learn to activate the screws of the actuators correctly to cover some distance. Furthermore there were obstacles placed in the environment. Thus, the AHHS-controller had to react on the input of the distance sensors as a next step. The fitness could be increased, on the one hand, by moving (standing at the wall with activated screws was not rewarded) and, on the other hand, by gaining distance from the starting point.



(a) all runs (n=8); circle: median, thick bar: quartiles, thin bar: min, max



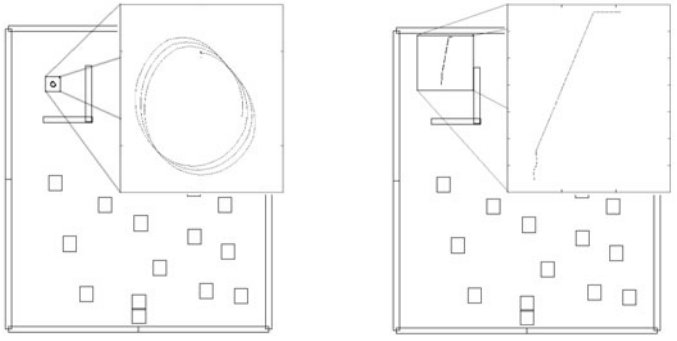
(b) run 2



(c) run 6

Fig. 3. Fitness progress of the best individual per generation over 50 generations. Arrows in (b) and (c) mark generations from which trajectories of the best individual are plotted in Fig. 4.

Eight runs were performed with the settings described in section 2.4. When evaluating all eight runs, jumps of the median of the maximum fitness are observed at generation 6 and 20 (Fig. 3(a)). Exemplarily, run 2 (Fig. 3(b)) shows this stepwise evolution. In this run, the controller activates only one screw such that the robot circulates around the starting point during the first 20 generations (Fig. 4(a)) or no actuator is activated at all. For the next 10 generations the robot drives a straight line, but the controller has not learned to react on



(a) run 2, gen. 6, ind. 2 (b) run 2, gen. 22, ind. 7

(c) run 2, gen. 50, ind. 14 (d) run 6, gen. 50, ind. 19

Fig. 4. Trajectories of the robot of run 2 (a, b and c) and 6 (d); gen.: generation number, ind.: number of the individual in the population.

sensory input and collides with the wall (Fig. 4(b)). After 50 generations in run 2 the sensor inputs lead to turning the robot and to wall following behaviour (Fig. 4(c)). In run 6 the evolution led to stronger turning induced by sensory input and therefore to a avoidance behaviour (Fig. 4(d)). This behaviour reached the overall maximum of the fitness value of 23.47 (Fig. 3(c)).

Post evaluation of the hormones and rules revealed the mechanism of the controller that steers the fittest robot. As depicted in Fig. 5(a) three hormones are responsible for the motion behaviour of the robot: Two hormones are produced and reach different equlibria ($H0 \rightarrow 9$, $H2 \rightarrow 84$, see Fig. 5(b)) and activate the screws for straight driving by different activation factors. Straight driving requires symmetric activation of the screws. As hormone 0 and hormone 2 have very different set points, AE had to precisely adapt the dependent dose and fixed dose (see 2.3). Sensor 3 reports values below the threshold ($S3 < 97$) at open space. Thus, it always triggers extensive secretion of hormone 7. Near obstacles the sensor value exceeds the limit of 97 units, therefore the production of hormone 7 ceases. As soon as hormone 7 falls below a threshold of 142 the left screw gets deactivated and the robot turns away from the wall. The evolved

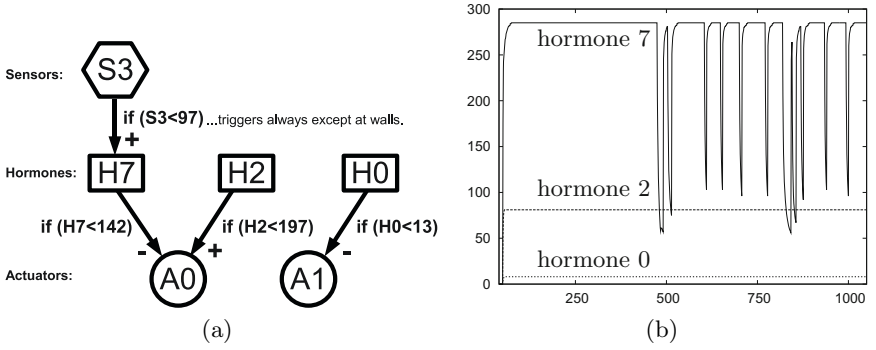


Fig. 5. (a) Schematic overview of the sensor-hormone-actuator interaction in the wall follower controller (see Fig. 4(c)). (b) Values of the three participating hormones

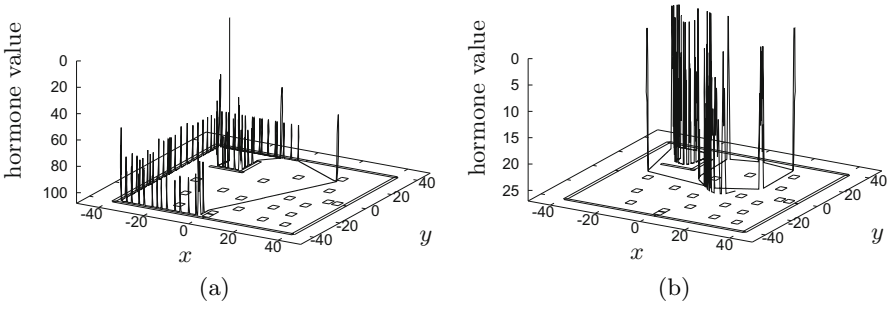


Fig. 6. The value of the critical hormone in (a) the wall follower controller (compare Fig. 4(c)) and (b) the wall avoider controller (compare Fig. 4(d))

controller is already a complex network of 3 hormones and 4 rules. The dynamics of the hormone concentrations which steer the wall follower and wall avoider are plotted in Fig. 6.

4 Conclusions

Our aim is to evolve robot controllers that regulate a swarm of robots which connect autonomously in various body shapes. In addition the controllers should coordinate the body movements of the connected organism. As a first approach towards this challenging task, we developed a framework for AE and present here first results which suggest that the AHHS controller is successfully adaptable by AE. For coordinated body movement of multi-modular robotic organisms, hormone-inspired controllers were suggested in [12]. In contrast to our AHHS controller, these hormones are implemented via simple message transfer instead of concentrations of molecules in a fluid. Other studies used hormone-like gradients to aggregate complex body forms starting from single simple modules [13].

In contrast to those approaches we focus on mimicking fluid diffusion processes, as it is found in real organisms.

Finally, as our AHHS controller is not fixed in size and in complexity (except for limitations in computing time and memory), we interpret that our AE offers almost open ended evolution. In our studies, a simple AHHS produced already interesting and different motion behaviours: circling, wall following, wall avoiding. The post evaluation showed that only a fraction of the resources were used to accomplish the task. Further tests will reveal the possibilities of the controller in more complex tasks.

References

1. Marocco, D., Nolfi, S.: Origins of communication in evolving robots. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D. (eds.) SAB 2006. LNCS (LNAI), vol. 4095, pp. 789–803. Springer, Heidelberg (2006)
2. Floreano, D., Nolfi, S., Mondada, F.: Co-Evolution and Ontogenetic Change in Competing Robots. In: Patel, M., et al. (eds.) Advances in the Evolutionary Synthesis of Intelligent Agents. MIT Press, Cambridge (2001)
3. Webb, B.: Can robots make good models of biological behaviour? Behavioral and Brain Sciences 24, 1033–1050 (2001)
4. Dorigo, M., Trianni, V., Sahin, E., Gro, R., Labella, T.H., Baldassarre, G., Nolfi, S., Mondada, F., Deneubourg, J.L., Floreano, D., Gambardella, L.M.: Evolving Self-Organizing Behaviors for a Swarm-bot. Autonomous Robots, special Issue on Swarm Robotics 17, 223–245 (2004)
5. Rechenberg, I.: Evolutionsstrategie 1994. Frommann Holzboog (1994)
6. Schmickl, T., Crailsheim, K.: Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm. Autonomous Robots 25, 171–188 (2008)
7. SYMBRION: Project website (2008), http://www.symbrion.eu/
8. REPLICATOR: Project website (2008), http://www.replicators.eu/
9. Delta 3D: Open-source gaming and simulation engine project website, http://www.delta3d.org/
10. Schmickl, T., Crailsheim, K.: Modelling a hormone-based robot controller. In: MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling (2009)
11. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Natural Computing Series. Springer, Heidelberg (2003)
12. Shen, W.M., Salemi, B., Will, P.: Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots. IEEE Transactions on Robotics and Automation 18, 700–712 (2002)
13. Stoy, K.: How to construct dense objects with self-reconfigurable robots. In: Christensen, H. (ed.) European Robotics Symposium 2006. STAR, vol. 22, pp. 27–37. Springer, Heidelberg (2006)

Functional and Structural Topologies in Evolved Neural Networks

Joseph T. Lizier[1,2], Mahendra Piraveenan[1,2], Dany Pradhana[1], Mikhail Prokopenko[1], and Larry S. Yaeger[3,⋆]

[1] CSIRO Information and Communications Technology Centre, Locked Bag 17, North Ryde, NSW 1670, Australia
[2] School of Information Technologies, The University of Sydney, NSW 2006, Australia
[3] School of Informatics, Indiana University, 919 E. 10th St., Bloomington, IN 47408, USA
mahendra.piraveenan@csiro.au

Abstract. The topic of evolutionary trends in complexity has drawn much controversy in the artificial life community. Rather than investigate the evolution of overall complexity, here we investigate the evolution of topology of networks in the Polyworld artificial life system. Our investigation encompasses both the actual structure of neural networks of agents in this system, and logical or functional networks inferred from statistical dependencies between nodes in the networks. We find interesting trends across several topological measures, which together imply a trend of more integrated activity across the networks (with the networks taking on a more "small-world" character) with evolutionary time.

1 Introduction

The nature of evolutionary trends in complexity has been subject to much debate [1], with interest surrounding whether the evolutionary growth in complexity of organisms in the natural world is the outcome of natural selection or some sort of random walk [2,3]. Indeed, this question has been explored in artificial life systems: e.g. previous work with Polyworld has demonstrated that evolution can and does select for increased complexity in a driven fashion in *some* circumstances, but also selects for complexity stability under other conditions [4,5].

Here, our interest lies not so much in the evolution of (any particular measure of overall) complexity, but rather the manner in which the *topology* of neural networks adapt under evolutionary pressure. Specifically, we investigate the topology of neural networks of agents in the Polyworld artificial life system. We examine both the actual structure of these networks, and their logical structure.

⋆ The authors thank the sponsors of the Guided Self-Organisation Workshop 2008 (GSO-2008), who partially supported this work: the Australian Research Council's Complex Open Systems Research Network (COSNet) and Research Network in Enterprise Information Infrastructure (EII), The University of Sydney, and CSIRO Complex Systems Science and ICT Centre.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 140–147, 2011.

The logical structure of the neural networks is explored by inferring functional networks [6,7] from statistical dependencies between the time series of each node in the underlying structural network. Here, we use mutual information [8] and transfer entropy [9] to measure the statistical dependencies between the neurons. We then examine the trends in several measures of the topology of the structural and functional networks with respect to evolutionary time: in particular, we measure the assortativity, modularity, clustering coefficient and closeness of the networks. We find several interesting trends in the topologies, with the trends in the structural and transfer entropy-based functional networks being most similar. These networks become more non-assortative, less modular but more clustered, and adopt shorter average path lengths with evolutionary time. These trends are significant in that they imply the networks are taking on a more "small-world" [10] character over evolutionary time.

2 Polyworld

Polyworld [11] is a computational ecology evolving populations of haploid agents, each using a suite of primitive behaviors (move, turn, eat, mate, attack, light, focus) under continuous control of an Artificial Neural Network (ANN) employing summing and squashing neurons with synapses that adapt via Hebbian learning. The wiring diagram of the ANN is encoded in the organism's genome, via a statistical description of the number of neural groups of excitatory and inhibitory neurons, synaptic connection densities, ordered-ness of connections, and learning rates. Input to the ANN consists of pixels from a rendering of the scene from each agent's point of view, like light falling on a retina. The agent morphologies are simple and fixed, but agents' interactions with the world and each other are fairly complex, as they replenish energy by seeking out and consuming food or by killing and eating other agents. They reproduce when two collocated agents simultaneously express their mating behaviors, using a number of crossover points and a mutation rate that are also contained in the parental genomes [11].

Bounds on the agent population, both high and low, are maintained by altering the energy consumption of the agents (as in [5]). As the population approaches the upper bound, the amount of energy depleted by all agent behaviors, including neural activity, is increased in a continuous fashion. Reciprocally, as the agent population approaches the minimum, energy depletion is decreased, and agent lifespans may be artificially extended.

The simulation is initially seeded with a uniform population of agents that have the minimum number of neural groups and a nearly minimal number of neurons and synapses. While predisposed to some potentially beneficial behaviors, such as running towards food (green) and away from aggression (red; see [11] for details on color use in Polyworld), these seed organisms are not a viable species. I.e., without evolution they cannot sustain their numbers through their reproductive behaviors and will inevitably die out.

As simulations progress both the structural architecture of the ANNs and the activation of every neuron at every time step are recorded for every agent.

Here we use these neural activation recordings to determine functional networks for each agent and compare functional network characteristics to the underlying structural network characteristics.

3 Inferring Functional Networks

Two remote neural nodes are defined to be functionally connected where they exhibit statistical dependence in time [6,7]. The *nodes* considered could be voxels in BOLD recordings (e.g. [7]), or neurons in an artificial neural network (as are used here). A functional network is then formed from a set of functional connections. Inferring functional networks from time-series of node states therefore involves two distinct steps: i. making some measure of the statistical dependence or closeness between each node pair, then ii. deciding whether each closeness value should constitute a link between the node pair. The closeness measure and the inferred links can be either directional or undirectional.

Functional networks may be used to infer the underlying structural network where this is unknown. More importantly, functional networks provide insight into the *logical* structure of the network and how this changes as a function of network activity (regardless of whether the underlying structure is known).

In this work, we use information-theoretical measures [8] for the closeness of each pair X and Y. The **mutual information** between X and Y measures the average reduction in uncertainty about x (or entropy H of x) that results from learning the value of y, or vice versa:

$$I(X;Y) = \sum_{x,y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}. \tag{1}$$

In this way, $I(X;Y)$ is a symmetric measure of the common information between X and Y. Though it has been previously used to measure directed information transfer from one variable to another, this is not valid: it is a symmetric measure of *statically shared* information (which is useful in its own right).

Alternatively, the **transfer entropy** [9] was introduced as a *directed* measure of *dynamic* information transfer from one variable to another. It quantifies the information provided by a source node about a destination's next state that was not contained in the past of the destination. Specifically, the transfer entropy from a source node Y to a destination X is the mutual information between the previous state of the source y_n and the next state of the destination x_{n+1}, *conditioned* on the past k states of the destination $x_n^{(k)}$:

$$T_{Y \to X}(k) = \sum_{x_{n+1}, x_n^{(k)}, y_n} p(x_{n+1}, x_n^{(k)}, y_n) \log_2 \frac{p(x_{n+1}|x_n^{(k)}, y_n)}{p(x_{n+1}|x_n^{(k)})}. \tag{2}$$

The transfer entropy may be measured for any two time series X and Y and is always a valid measure of the predictive gain from the source, but only represents physical information transfer when measured on a causal link [12].

Here, we compute functional networks for each agent from the Polyworld simulation using both mutual information and transfer entropy as separate measures of closeness. The continuous activation levels are first discretised in four levels, and a history length $k = 1$ is used for the transfer entropy (this renders it more towards an inference of causal effect than information transfer [13,12]).

Several options are then available for deciding whether each pair of areas should be considered functionally connected based on their closeness. One could assign links to a given number or percentage of pairs based on the largest closeness values, or could use an approach based on the statistical significance of the closeness measure, e.g. [14]. Here, the number of functional links was designed to match the proportion of links in the underlying structural network, and the largest such closeness values were assigned links. A (directed) link exists in the structural network between two neurons where the source neuron is an input to the target neuron. We consider both processing and input neurons in the functional network.

4 Network Topological Measures

Analysis of the *topology* of functional networks provides useful information about the dynamic behaviour of the network [7,14]. In this section, we introduce the measures of topology used to analyse the functional networks here. All were calculated using [15].

Assortativity is the tendency observed in networks where nodes mostly connect with similar nodes. Typically, this similarity is interpreted in terms of degrees of nodes. Assortativity has been formally defined as a correlation function of excess degree distributions and link distribution of a network [16,17]. The concepts of degree distribution $p(k)$ and excess degree distribution $q(k)$ for undirected networks are well known [17]. Given $q(k)$, one can introduce the quantity $e_{j,k}$ as the joint probability distribution of the remaining degrees of the two nodes at either end of a randomly chosen link. Given these distributions, the assortativity of an undirected network is defined as:

$$r = \frac{1}{\sigma_q^2} \left[\sum_{jk} jk \left(e_{j,k} - q\left(j\right) q\left(k\right) \right) \right], \tag{3}$$

where σ_q is the standard deviation of $q(k)$. Assortativity distributions can be constructed by considering the *local assortativity* of all nodes in a network [18].

Closeness centrality of a node v is defined as the mean geodesic distance (shortest path length) between the node and all other nodes in the network [19]. (Sometimes the quantity is inverted so that the nodes which are 'most central' to the network G would get higher values). Closeness centrality is formally defined as $C^C\left(v\right) = \sum d_G(v,t)$ where $v \neq t$ and $d_G(v,t)$ is the shortest path distance between nodes v and t.

Network **modularity** is the extent to which a network can be separated into independent sub-networks. Formally [20], modularity quantifies the fraction of

links that are within the respective modules compared to all links in a network. [20] introduces an algorithm which can partition a network into k modules and measure the partition's modularity Q. The measure uses the concept that a good partition of a network should have a lot of within-module links and a very small number of between-module links. The modularity can be written as:

$$Q = \sum_{s=1}^{k} \left[\frac{l_s}{L} - \left(\frac{d_s}{2L} \right)^2 \right], \tag{4}$$

where k is the number of modules, L is the number of links in the network, l_s is the number of links between nodes in module s, and d_s is the sum of degrees of nodes in module s. To avoid getting a single module in all cases, this measure imposes $Q = 0$ if all nodes are in the same module or nodes are placed randomly into modules.

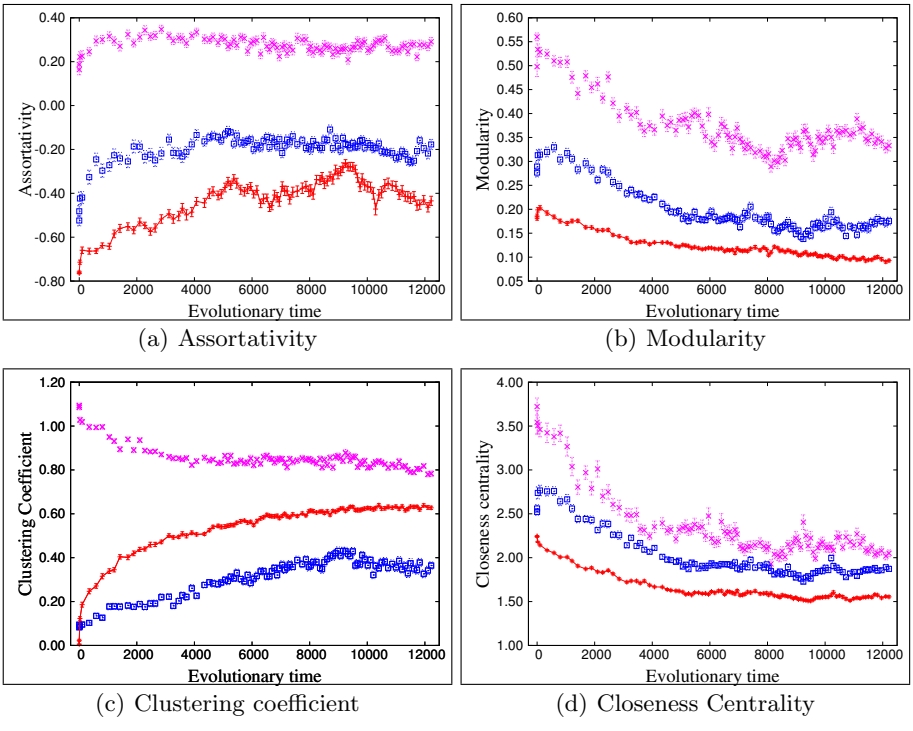
The **clustering coefficient** of a node characterizes the density of links in the environment closest to a vertex. Formally, the clustering coefficient C of a node is the ratio between the total number y of links connecting its neighbours and the total number of all possible links between all these z nearest neighbours [21]: $C = 2y/(z(z-1))$. The clustering coefficient $\overline{C}$ for a network is the average C over all nodes.

5 Results and Discussion

We constructed the functional networks for each agent, and evaluated each measure of network topology on these and the underlying structural networks (which had between 13 and 159 neurons, and 52 on average). We then averaged each measure over sets of 100 sequential agents ordered by birth. The results are plotted with respect to evolutionary time in Fig. 1. Clearly, all measures reach a relatively steady state within 5000 – 12000 steps in evolutionary time. This aligns with previous studies of trends in the complexity of the neural networks in Polyworld [5] where the complexity is driven upwards over the initial 5000 or so steps of evolution before the agents find a "good enough" solution. At this point the drive for evolutionary change somewhat stagnates, as is reflected in the steady state of the measures here.

In general, the transfer entropy-inferred functional networks show similar trends to the structural networks across all measures. Interestingly, the transfer entropy-inferred functional networks had a slightly smaller overlap (mean $17.6 \pm 0.1\%$) with the underlying structural networks than the mutual information-inferred functional networks (mean $19.1 \pm 0.1\%$). It is possible that the transfer entropy performs better at inferring the general interaction structure between modules or regions in the structural network (thereby capturing the general topological trends) without necessarily inferring the precise links any better.

As shown in Fig. 1(a), the structural networks tend to exhibit a negative assortativity: this is not surprising as it is a known general characteristic of biological networks evolved under external pressure [22]. This is because negative assortativity supports connectivity between diverse elements in the network, an important feature for producing complex behaviour. Unsurprisingly also, the mutual



(a) Assortativity (b) Modularity

(c) Clustering coefficient (d) Closeness Centrality

Fig. 1. Trends in structural and functional networks versus evolutionary time. Measures are plotted for structural networks (red line), mutual information-inferred functional networks (violet ×), and transfer entropy-inferred functional networks (blue □). Error bars indicate the standard error of the mean.

information-inferred networks exhibit positive assortativity (since mutual information is maximised for similar elements), while the transfer entropy-inferred networks exhibit negative assortativity (since transfer entropy is minimised for similar elements). More interestingly, the structural and transfer entropy-inferred networks become more neutrally assortative over time (i.e. less negatively assortative). While this may seem surprising, it is possibly an artifact of the elements in the network becoming more closely coupled as they evolve and therefore become more similar, or perhaps reflects the increased clustering occurring over evolutionary time.

Fig. 1(b) and Fig. 1(c) show that the structural and transfer entropy-inferred networks become less modular but more clustered as they evolve. This is not a contradiction: it indicates that the boundaries between modules are becoming blurred with previously separated modules becoming more strongly clustered both *within* themselves and *across* each other (i.e. finding the right balance between functional integration and segregation to give rise to complex behaviour). The mutual information-inferred networks however exhibit a decrease in

clustering coefficient. Again, this seems to be a relic of the mutual information measure being maximised for similar elements: stronger coupling across clusters in the underlying network is likely to diversify the activity of previously similar nodes, thereby reducing clustering in this functional network.

Finally, Fig. 1(d) shows that the closeness centrality is reduced with evolutionary time for all networks. Given the previous results, this is unsurprising as all imply diversification of connectivity across the network with evolutionary time. In fact, taken together these results (in particular the higher clustering and lower shortest path lengths) suggest that the networks are becoming more small-world [10] with evolutionary time. Again this is unsurprising but significant, since the same effect is observed in many natural systems (including biological cortical networks and networks optimised for complexity [23], as well as functional networks inferred from neural networks in [14]) due to the advantages bestowed by this property. Importantly though, recall that all measures reach a steady state here: the neural networks do not continually improve on these desirable features, but stop developing once a good enough solution is found.

6 Conclusion

We have measured functional networks to represent the logical activity of neural networks of agents in the Polyworld artificial life system. Topological analysis of these functional networks, and the underlying structural networks, revealed clear trends with evolutionary time. The structure and activity in the networks becomes more integrated over time, as may be expected in the evolution of complex distributed processes. In particular, both the structural and functional networks take on more of a small-world character as the evolution progresses.

Our results also showed interesting differences between the use of mutual information and transfer entropy in inferring functional networks. The transfer entropy-inferred functional networks have topological trends more similar to those of the underlying structural networks, and also provided more intuitive insights into network activity.

In extending this work, it would be desirable to evaluate the statistical significance of the trends observed here. One method for doing this would be to contrast the results here (where evolution is driven by genetic mixing) with those produced by passive genetic drift (along the same lines as the comparison of trends in complexity in [5]).

References

1. Bedau, M.A.: The evolution of complexity. In: Barberousse, A., Morange, M., Pradeu, T. (eds.) Mapping the Future of Biology. Boston Studies In The Philosophy Of Science, vol. 266, pp. 111–130. Springer, Netherlands (2009)
2. Gould, S.J.: The evolution of life on earth. Scientific American 271(4), 62–69 (1994)
3. Maynard Smith, J.: Time in the evolutionary process. Studium Generale 23, 266–272 (1970)

4. Yaeger, L., Sporns, O.: Evolution of neural structure and complexity in a computational ecology. In: Rocha, L.M., Yaeger, L.S., Bedau, M.A., Floeano, D., Goldstone, R.L., Vespignani, A. (eds.) Proceedings of the Tenth International Conference on Simulation and Synthesis of Living Systems (ALifeX), Bloomington, Indiana, USA, pp. 330–336. MIT Press, Cambridge (2006)
5. Yaeger, L., Griffith, V., Sporns, O.: Passive and driven trends in the evolution of complexity. In: Bullock, S., Noble, J., Watson, R., Bedau, M.A. (eds.) Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems (ALifeXI), Winchester, UK, pp. 725–732. MIT Press, Cambridge (2008)
6. Friston, K.J.: Functional and effective connectivity in neuroimaging: A synthesis. Human Brain Mapping 2, 56–78 (1994)
7. Honey, C.J., Kotter, R., Breakspear, M., Sporns, O.: Network structure of cerebral cortex shapes functional connectivity on multiple time scales. Proceedings of the National Academy of Sciences 104(24), 10240–10245 (2007)
8. MacKay, D.J.: Information Theory, Inference, and Learning Algorithms. Cambridge University Press, Cambridge (2003)
9. Schreiber, T.: Measuring information transfer. Phys. Rev. Lett. 85(2), 461–464 (2000)
10. Watts, D.J., Strogatz, S.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
11. Yaeger, L.S.: Computational genetics, physiology, metabolism, neural systems, learning, vision and behaviour or polyworld: Life in a new context. In: Langton, C.G. (ed.) Proceedings of the Artificial Life III Conference, Santa Fe, NM, USA, pp. 263–298. Addison-Wesley, Reading (1994)
12. Lizier, J.T., Prokopenko, M.: Differentiating information transfer and causal effect. Euro. Phys. J. B. 73(4), 605–615 (2010)
13. Lizier, J.T., Prokopenko, M., Zomaya, A.Y.: Local information transfer as a spatiotemporal filter for complex systems. Phys. Rev. E 77(2), 26110 (2008)
14. Bettencourt, L.M.A., Stephens, G.J., Ham, M.I., Gross, G.W.: Functional structure of cortical neuronal networks grown in vitro. Phys. Rev. E 75(2), 21915 (2007)
15. Sporns, O., Rubinov, M., Kötter, R.: Brain connectivity toolbox (2009), http://www.brain-connectivity-toolbox.net/
16. Newman, M.: Assortative mixing in networks. Phy. Rev. Lett. 89(20), 208701 (2002)
17. Newman, M.: Mixing patterns in networks. Phy. Rev. E 67(2), 26126 (2003)
18. Piraveenan, M., Prokopenko, M., Zomaya, A.Y.: Local assortativeness in scale free networks. Euro. Phys. Lett. 89(2), 28002 (2008)
19. Shimbel, A.: Structural parameters of communication networks. Bulletin of Mathematical Biology 15(4), 501–507 (1953)
20. Alon, U.: An Introduction to Systems Biology: Design Principles of Biological Circuits. Chapman & Hall/Crc Mathematical and Computational Biology Series. Chapman & Hall/CRC (July 2006)
21. Dorogovtsev, S., Mendes, J.: Evolution of Networks: From Biological Nets to the Internet and WWW. Oxford University Press, Oxford (2003)
22. Solé, R.V., Valverde, S.: Information theory of complex networks: On evolution and architectural constraints. In: Ben-Naim, E., Frauenfelder, H., Toroczkai, Z. (eds.) Complex Networks. Lecture Notes in Physics, vol. 650, pp. 189–207. Springer, Heidelberg (2004)
23. Sporns, O., Kötter, R.: Motifs in brain networks. PLoS Biology 2(11), e369 (2004)

Input from the External Environment and Input from within the Body

Filippo Saglimbeni and Domenico Parisi

Institute of Cognitive Sciences and Technologies, National Research Council, Rome
{filippo.saglimbeni,domenico.parisi}@istc.cnr.it

Abstract. Behaviour responds to both input from the external environment and input from within the organism's body. Input from the external environment has mainly the function to regulate the execution of the organism's activities while input from the body is used to decide which activity to execute. We evolve artificial organisms which to survive and reproduce have to both eat food and drink water in equivalent quantities and therefore at any given time they have to decide whether to look for food or water. We show that in some environments the appropriate behaviour can evolve with no need for the organism's brain to know the current level of energy and water in the body while in other environments the brain needs this information from the body in the form of hunger and thirst. We discuss how the body and the body's interactions with the brain are part of the overall adaptive pattern of an organism and must co-evolve with brain and behaviour.

1 Introduction

To survive and reproduce minimally complex organisms must be able to both execute effectively a number of diverse activities and to decide which activity to execute at any given time. These are two distinct abilities. Consider an organism that to survive has to both eat and drink. The organism's body includes a store of energy and a store of water and at each time step a fixed quantity of energy and water is consumed to keep the organism alive - if any of the two stores reaches zero level the organism dies. To remain alive the organism must be able to find food (energy) and water in the environment. The organism must also look for food or water when the level of either is low. Clearly, the individuals that survive must possess both abilities.

We call these two components of the adaptive pattern of organisms the *cognitive* (or tactical) component and the *motivational* (or strategic) component. Most research aimed at constructing artificial organisms that resemble real organisms is dedicated to studying the cognitive component of behaviour, that is, to endowing artificial organisms with the ability to execute a single activity aimed at some specific goal, although this single activity may be a complex one with a hierarchical structure of sub-abilities. The cognitive component of behaviour can be interpreted as the ability to respond to stimuli from the environment with the appropriate movements; but the behaviour of organisms is also caused by the internal states of the organism's body or brain. In fact, the sight of food should induce a behaviour of approaching and eating the food only if the organism is hungry. Otherwise, the food should be ignored. This simple example indicates the importance of the organism's internal states in determining the organism's behaviour.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 148–155, 2011.

Recently research sought to capture the motivational and emotional aspects of behaviour with artificial organisms [1,2,3,4,5,6,7]. Studying behaviour by constructing embodied artificial organisms (robots) should facilitate an examination of the motivational and emotional aspects of behaviour since motivation and emotion appear intrinsically linked to the body beyond the brain and to the interactions between the body and the brain [4,8,9,10,11]. Robots have an "external body" (size, shape, sensory and motor organs) but not an "internal body" with its organs and systems. The study of motivation and emotion requires the development of both an external and an internal robotics [12].

We might say that organisms live in, and have to adapt to, two environments: the environment which is outside their body and the "internal environment" constituted by their own body. However, the two environments have a critical difference. While the external environment is what it is mainly independent from the organism (except for human technology), the internal environment clearly is part of the overall adaptive pattern of the organism and it evolves with the organism's behavior [13].

In this paper we will describe a number of simple simulations showing that the existence of a communication channel between the energy and water stores inside the organism's body and the organism's brain can be adaptive in some stereotypical environments but not in all environments. In some environments organisms may need to feel hungry and thirsty to survive but hunger and thirst are adaptations and they may not be particularly useful in other environments.

2 The Simulation Scenario

Our organisms are a simulated version of the Khepera robot [14] and we use the Evorobot* simulation tool (developed by Stefano Nolfi; cf. http://laral.istc.cnr.it/evorobotstar/). They have a cylindrical body, sensors with which they can detect food and water tokens, and two wheels that can be moved independently at different velocities. The organisms have energy and water stores with a level that can go from 1 (full store) to 0 (empty store). When they are born both stores are completely filled up but a fixed amount of energy and water is consumed at each time step.

The simulated organism lives in a walled environment of 1000x1000 pixels and its body occupies a circle of 75 pixels of diameter. When the organism's body reaches the wall, its orientation is changed randomly. The environment contains food and water tokens each of witch occupies a circle of 30 pixels. When the center of the organism's body enters in a token circle, the token disappears (and is replaced by a new token in another randomly chosen location) and the organism's relative body level is increased.

The entire lifetime of an organism is made up of 10 epochs each lasting 1500 time steps. However, most of the time, the actual lifetime is shorter than that because an epoch is terminated if either the energy or water store of the organism goes to zero. At the beginning of each epoch the organism is placed at the center of the environment with a randomly chosen orientation.

The behaviour of the organisms is controlled by a neural network with 4 input (sensory) units, 2 output (motor) units and 4 hidden units. Each of the 4 input units sends its connections to all hidden units, and each hidden unit sends its connections to each output unit. In the simulations in which the organism's nervous system is informed of

the levels of energy and water in its body, the organism's neural network has 2 additional sensory units that send connections to all 4 hidden units. We will call these units "motivational units" (hunger and thirst units). 2 of the 4 sensory units detect the food tokens and the other 2 detect the water tokens; in each pair one unit gets activated by tokens seen on the right (RU for right unit), and the other one by tokens seen on the left (LU for left unit), according to the following expressions:

$$RU(d,\phi) = K\Big[A+B\,Log\Big(\frac{1}{d^2}\Big)e^{-\frac{(\phi+60^o)^2}{2\sigma^2}}\Big],\; LU(d,\phi) = K\Big[A+B\,Log\Big(\frac{1}{d^2}\Big)e^{-\frac{(\phi-60^o)^2}{2\sigma^2}}\Big].$$

When a food/water token appears in the visual field of the organism, the activation levels of the corresponding sensory units vary with the logarithm of the inverse square distance, d, of the token from the organism; the activation depends also on the angular position of the token in the organism's visual field, ϕ - the left and right half-fields are set to be oriented, respectively, 60^o to the left and 60^o to the right with respect to the frontal direction; σ determines the eye angular view spread and its value is 45^o. K, A and B are constant values set up to ensure activation spans the interval $[0, 1]$ ($A = 1.596$, $B = 0.110$, $K = 0.75/Log(N)$ in environments (1), (2), (3), and $K = 0.5/Log(N)$ in Env. (4), where N is the total number of tokens in the environment (see below for environment descriptions)). Each of the activation values of the two output unit determines (linearly) the separate speed of the corresponding wheel and therefore the trajectory followed by the organism. The 2 *motivational* units, when present, are internal sensory units informing the neural controller of the level of energy (*hunger* unit) and water (*thirst* unit), in the organism's body. The activation value of each of these units maps linearly the level of the corresponding resource (1 for full store, 0 for empty store).

Each simulation starts with a population of 100 randomly generated organisms. At the end of the 10 epochs comprising their life, each organism is assigned a fitness which is simply the total duration (number of time steps) of its life. The individuals which eat food and drink water in sufficient and comparable quantities live longer in each epoch and therefore are more likely to have offspring - the 20 robots with highest fitness are selected for reproduction. Each robot generates 5 offspring inheriting the same genome of their (single) parent, with the addition of random mutations (each one of the bits of the genome has a 4% probability of being mutated). Each simulation lasts for 1000 generations and is repeated 10 times starting from randomly generated organisms.

We have run four different simulations in four different stereotypical environments (for other details see Tab. 1). Env. (1) contains 5 food tokens and 5 water tokens. Env. (2) contains 5 food tokens and only 1 water token. Env. (3) is "seasonal": it contains 5 food tokens and only 1 water token in 5 of the 10 epochs of an individual's lifetime, and 5 water tokens and only 1 food token in the other 5 epochs. In all these 3 environments the tokens are randomly distributed. Env. (4) contains 3 food tokens and 3 water tokens but the tokens are distributed in patches, with all the food tokens located inside a square of 60 pixels of side and the same for the water tokens, while the centers of the two patches are at a distance of 600 pixels.

We have evolved two different populations in each of the four environments. The organisms of one population (Sim for "simple") do not have the motivational circuit while the organisms of the other population (Mot) do have this circuit.

3 Results

3.1 Fitness

Figure 1 and Table 2 show the fitness distributions of the 1000 individuals of the final generation of the 10 replications of the evolution in all four environments and their mean and standard deviations, separately for the Sim and the Mot organisms (the Mem organisms will be discussed later in the section "Motivation as memory").

We see that in environments (1) and (2) the two populations reach comparable levels of fitness, whereas in environments (3) and (4) the Mot populations perform better: the presence of the information coming from the body stores correlates with higher adaptive skills in the environments (3) and (4), but not in environments (1) and (2).

Balanced environment (same quantity of food and water tokens): the organisms can adapt to this environment by developing a simple behaviour which consists in approaching whatever token is closest, regardless of it is a food or a water token. This behaviour ensures both foraging efficiency and diet balancing and does not require the knowledge of the current bodily levels of energy and water.

Unbalanced environment (food is five times scarcer than water): in this environment too it is possible for the organisms to develop an effective and balanced behaviour with no need for their nervous system to be informed about the current bodily levels of energy and water: they can simply evolve a tuned *preference* for food.

Seasonal environment (food tokens are five times scarcer than water tokens in half of the seasons and the opposite is true in the other half of the seasons): in a seasonal environment the behavioural strategies of the organisms living in a "static" environment like (1) and (2) are suboptimal, because they are not capable of coupling with the ever

Table 1. Environment features (*seasonal* environment: outside the brackets one season, inside brackets the opposite season)

	Env. (1): *balanced*	Env. (2): *unbalanced*	Env. (3): *seasonal*	Env. (4): *patched*
Living cost per cycle	0.040	0.040	0.025	0.025
Minimum lifetime	250	250	400	400
Energy per token	0.2	0.4	0.4	0.03
# of food tokens	5	1	1(5)	3
# of water tokens	5	5	5(1)	3

Table 2. Fitness data in all the four environments at last generation of evolution. "Ave" is the population's mean fitness, "Best" is the best individual's fitness.

		balanced	*unbalanced*	*seasonal*	*patched*
Ave	Sim	302 ±2	330 ±5	574 ±60	480 ±3
	Mot	299 ±2	329 ±5	685 ±16	485 ±9
	Mem	311 ±2	366 ±5	672 ±26	489 ±9
Best	Sim	353 ±2	435 ±8	801 ±118	561 ±3
	Mot	350 ±2	443 ±10	1064±32	614 ±32
	Mem	370 ±3	504 ±9	987 ±71	581 ±8

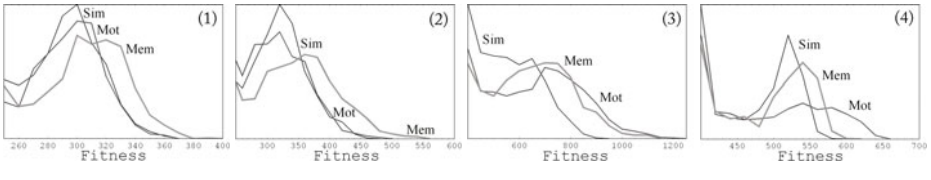


Fig. 1. Fitness distributions of the populations of the last generation of evolution in all the four environments (from left to right): (1) *balanced*, (2) *unbalanced*, (3) *seasonal*, (4) *patched*. Comparison between Sim, Mot and Mem populations.

changing environmental conditions (remember that our organisms haven't got any type of "season sensor"). In Env. (3) the communication channel between the body stores and the brain results a strong adaptive tool (see Fig. 1, and Tab. 2, third column), consenting our organisms to *counterbalance* the environmental biases of seasons by going after food when energy in their body is low and going after water when water is low.

Patched environment (food and water are equally abundant but the tokens are distributed in two separate patches, one with food tokens and the other one with water tokens): in this environment too the organisms cannot simply go after the token which is closest to them like the organisms living in Env. (1) because if an organism happens to be in a food patch this behavioural strategy would imply eating a lot of food but possibly running out of water and dying, and vice versa if the organism finds itself in a water patch. For the organisms living in Env. (4) it is advantageous to feel hunger and thirst in order to be able to *abandon* a food patch if they are thirsty and a water patch if they are hungry (see next section for more details).

3.2 Experimental Tests

To test this interpretation of the fitness results we have tested the individuals of the last generation in each of the four simulations with and without motivational units in controlled, "experimental", conditions, identical for all individuals. We examined the behaviour of each individual in a situation in which the individual is exposed to a single food and water token at the same time, with the two tokens located one at 45^o to the left and the other at 45^o to the right with respect to the organism facing orientation (in all conditions we exchanged the position of the food and water tokens). In different conditions the food and water tokens are located at 5 different distances from the organisms, where the ratio of the distances from the organism of the two tokens is varied between 1 (equal distances) to 5 (one token five times closer to the organism than the other token). Furthermore, for the organisms which receive information from the body (hunger and thirst), in each condition energy and water can have the following pairs of levels: 0.25/1, 0.33/1, 0.5/1, 0.5/0.5, 1/0.5, 1/0.33, 1/0.25, i.e., the organisms can have the same level of hunger and thirst (0.5/0.5) or they can be much more hungry than thirsty, or vice versa.

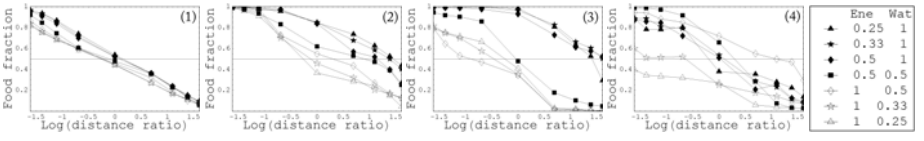


Fig. 2. Average *food choice fraction* as a function of the logarithm of the ratio between the distance of the food token and the distance of the water token from the organism in all the four environments (from left to right): (1) *balanced*, (2) *unbalanced*, (3) *seasonal*, (4) *patched*. The test is performed on the Mot individuals of the last generation of evolution for several (see legend) values of energy and water in the body.

The quantitative results of our analysis of the behaviour of the Mot organisms living in different environments are shown in Fig. 2, and can be summarized as follows:

Balanced environment: the organisms tend to go to the nearest token, regardless of it is food or water (negative slope of the curves) - this is not much affected by the levels of the two body stores (the curves are all close to each other).

Unbalanced environment: the organisms tend to prefer the food tokens, less abundant in their environment (the curves are shifted upwards). The organisms have learnt to use the information coming from the body: they increase their preference for food when they are more hungry than thirsty, and they decrease it when the opposite is true (the filled dots curves are above the empty dots ones); at full stores (1/1) the organisms show no preference for any of the two types of tokens when food is roughly 3 times more distant than water.

Seasonal environment: the organisms choose to go to the nearest token if they are equally hungry and thirsty (black filled squares) but the bodily state strongly biases their preference towards the more needed resource.

Patched environment: in this environment too, the levels of the two body stores influence the choice behaviour of the organisms in the adaptive way (filled dots curves mostly above the empty dots ones) even though the data are more noisy.

These results allow us to say that Mot individuals have learnt how to use the information arriving from within the body as this is a useful adaptation in environments (2), (3) and (4). This ability leads to an adaptive advantage in the *seasonal* and the *patched* environments, but not in the *unbalanced* environment (space precludes an in depth discussion of this point here). It is worth noting that, even if the difference in the average population's fitness between Sim and Mot organisms in the *patched* environment is not very great, the strategies they have evolved to survive are different. The Sim organisms developed a - not very efficient - "back and forth" strategy: they go straight towards the patch they see, and they spin when they don't see anything. In contrast, the Mot organisms show a sort of "restricted area search" (ARS) behaviour [15]: they remain in the patch they are in (eating or drinking) and they abandon it to reach the other one only when the relative body store is almost empty.

4 Motivation as Memory

There might be an alternative interpretation for our results, based on *memory* rather than *motivation*. The present state of the body might function, if it is communicated to the brain, as a sort of memory of what the organism has done recently. When the energy level is high and the water level is low, this means that the organism recently has eaten and not drunk and therefore it should drink rather than eat, and the opposite when the energy level is low and the water level is high. Since memory of recent behaviour is useful in environments (3) and (4) but not in environments (1) and (2), the existence of a motivational circuit results in higher fitness in environments (3) and (4) only.

To test this alternative interpretation we have added an explicit memory mechanism to the neural network of our organisms consisting of two parts. The network's hidden units are now *leaky* neurons and have fully *recurrent connections* [16]. A population of organisms endowed with this new neural network but without the motivational circuit has been evolved in all four environments.

The results show that in all four environments the organisms possessing the memory mechanism reach a higher fitness level compared to those without it (see Mem data

in Fig. 1 and Tab. 2). In other words, while the motivational circuit leads to a higher performance only in environments (3) and (4), the memory circuit leads to a higher performance in all four environments. This seems to indicate that memory and motivation are two distinct mechanisms, with separate effects on organism performance. The memory circuit has a positive influence on the cognitive component of organism behaviour, causing a more effective manner of approaching tokens in the environment and therefore being useful in all sorts of environments (data not shown). In contrast, the motivational circuit has a positive influence on the motivational component of the organisms' behaviour, leading to more effective "decisions" on whether to approach food or water and therefore being useful only in the particular environments in which such decisions are critical for survival, i.e., in our environments (3) and (4).

A further proof in favor of a distinction between memory and motivation is that evolved organisms endowed with both our memory circuit and our motivational circuit reach a higher level of performance in environments (3) and (4) with respect to both the organisms possessing only the memory circuit and the organisms possessing only the motivational circuit (data not shown). This clearly indicates that the two circuits have distinct functional roles and that in the appropriate environments these functional roles can have separate and additive beneficial influences on organism performance.

5 Discussion

Evolving a system that informs an organism's brain of the current state of the organism's body depends on the environment in which the organisms happen to live. All our organisms need to both eat and drink in more or less equal quantities in order to survive and have offspring. However, possession of a communication channel between body and brain that informs the brain of the current level of energy and water in the body is only advantageous in some environments. Examples of such environments are an environment in which food and water abundances change seasonally and an environment in which food and water are distributed in patches. In these environments it is critical for the organisms to evolve a motivational system that tells the brain how much energy and water is currently contained in the body so that behaviour can be determined by both input from the external environment and input from within the body.

It is interesting to note that while the external environment is given, the internal environment is not given but co-evolves with the brain. To adapt to the external environment means to develop the appropriate sensory organs and the appropriate neural processing system that allow the organisms to survive and reproduce in that environment. To survive in our *seasonal* and *patched* environments the organisms have to develop both a body that sends the appropriate input to the brain and a brain that responds appropriately to this input from their body.

We conclude by indicating two directions of future research. The role of an evolving body in the general process of adaptation can be studied in other ways. For example we could take into account the fact that the rate of consumption of energy and of water is not a given but is part of the entire adaptive pattern of the particular organism, and therefore can co-evolve with the rest of the organism, i.e., with its sensory organs, brain, and behaviour. A second direction of research concerns other aspects of competition between motivations (for a study of action selection in a social environment see [17]).

We are currently running simulations in which the organisms have two motivations: eating food and avoiding being captured by a predator. These simulations seem to indicate that there are two types of individuals which tend not to have offspring: individuals that are not very good at finding food (a tactical or cognitive problem) and individuals that are too afraid of the predator to look for food (a strategic or motivational problem).

References

1. Cañamero, L.: Modelling motivation and emotions as a basis for intelligent behaviour. In: Lewis Johnson, W. (ed.) Proceedings of the First International Symposium on Autonomous Agents, pp. 148–155. ACM Press, New York (1997)
2. Cañamero, L.: Emotion understanding from the perspective of autonomous research. Neural Networks 18, 445–455 (2005)
3. Cecconi, F., Parisi, D.: Neural networks with motivational units. In: Meyer, J.-A., Roitblat, H.L., Wilson, S.W. (eds.) From Animals to Animats 2, pp. 346–355. MIT Press, Cambridge (1993)
4. Pérez, C.H., Moffat, D.C., Ziemke, T.: Emotions as a bridge to the environment: On the role of body in organisms and robots. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D., et al. (eds.) SAB 2006. LNCS (LNAI), vol. 4095, pp. 3–16. Springer, Heidelberg (2006)
5. Montebelli, A., Herrera, C., Ziemke, T.: On cognition as dynamical coupling: An analysis of behavioral attractor dynamics. Adaptive Behavior 16(2-3), 182–195 (2008)
6. Parisi, D.: Motivations in artificial organisms. In: Tascini, G., Esposito, V., Zingaretti, P. (eds.) Machine Learning and Perception, pp. 3–19. World Scientific, Singapore (1996)
7. Ziemke, T.: On the role of emotion in biological and robotic autonomy. BioSystems 91, 401–408 (2008)
8. Cos-Aguilera, I., Cañamero, L., Hayes, M., Gillies, A.: Ecological integration of affordances and drives for behavior selection. In: Bryson, J.J., Prescott, T., Seth, A . (eds.) Modeling Natural Action Selection. Proceedings of the IJCAI 2005 Workshop, pp. 225–228 (2005)
9. Damoulas, T., Cos-Aguilera, I., Hayes, G.M., Taylor, T.: Valency for adaptive homeostatic agents: relating evolution and learning. In: Capcarrère, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., Timmis, J. (eds.) ECAL 2005. LNCS (LNAI), vol. 3630, pp. 936–945. Springer, Heidelberg (2005)
10. French, L.B., Cañamero, L.: Introducing neuromodulation to a Braitenberg vehicle. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 4199–4204. IEEE Press, Los Alamitos (2005)
11. McFarland, D., Spier, E.: Basic cycles, utility and opportunism in self-sufficient robots. Robotics and Autonomous Systems 20, 179–190 (1997)
12. Parisi, D.: Internal robotics. Connection Science 16, 325–338 (2004)
13. Acerbi, A., Parisi, D.: The evolution of pain. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 816–824. Springer, Heidelberg (2007)
14. Mondada, F., Franzi, E., Ienne, P.: Mobile robot miniaturization: A tool for investigation in control algorithms. In: Proceedings of the Third International Symposium on Experimental Robotics, Kyoto, Japan (1993)
15. Hills, T., Brockie, P.J., Maricq, A.V.: Dopamine and glutamate control Area-Restricted Search behaviour in Caenorhabditis elegans. Journal of Neuroscience 24, 1217–1225 (2004)
16. Elman, J.L.: Finding structure in time. Cognitive Science 14, 179–211 (1990)
17. Avîla-Garcia, O., Cañamero, L.: Using hormonal feedback to modulate action selection in a competitive scenario. In: Schaal, S., Ijspeert, A.J., Billard, S., Vijayakumar, S., Hallam, J., Meyer, J.-.A. (eds.) From Animals to Animats 8, pp. 243–252. MIT Press, Cambridge (2004)

Towards Self-reflecting Machines: Two-Minds in One Robot

Juan Cristobal Zagal and Hod Lipson

Computational Synthesis Laboratory, Mechanical and Aerospace Engineering,
Cornell University, Ithaca, NY 14853, USA
{jcz35,hod.lipson}@cornell.edu

Abstract. We introduce a technique that allows a robot to increase its resiliency and learning skills by exploiting a process akin to self-reflection. A robot contains two controllers: A pure reactive *innate* controller, and a *reflective* controller that can observe, model and control the *innate* controller. The reflective controller adapts the innate controller without access to the innate controller's internal state or architecture; Instead, it models it and then synthesizes filters that exploit its existing capabilities for new situations. In this paper we explore a number of scenarios where the innate controller is a recurrent neural network. We demonstrate significant adaptation ability with relatively few physical trials.

Keywords: Self-modeling, self-reflection, machine learning, evolutionary robotics.

1 Introduction

The process of self-reflection underlies much of humans' and primates' ability to adapt to vastly varying conditions with little or no physical experimentation. In this

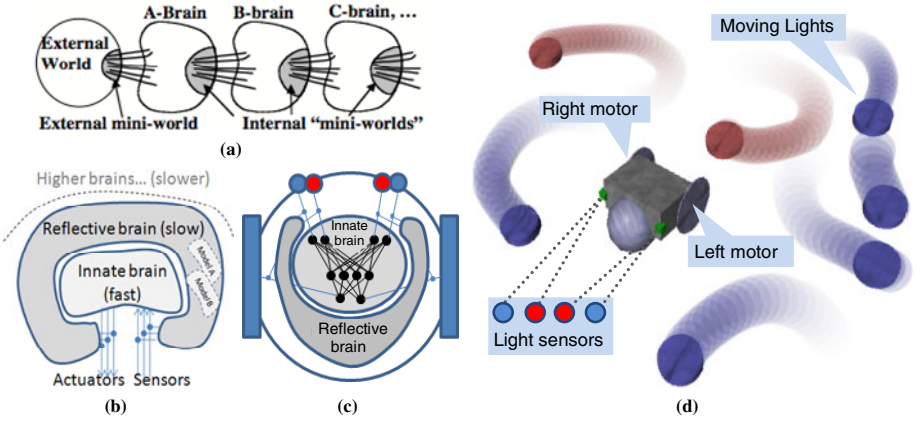


Fig. 1. (a) Minsky's brain chain, from [7]. (b) A proposal of nested brains architecture. (c) The robot contains an innate brain, and a reflective brain which can model and modulate the I/O of the innate brain. (d) Schematic of robot and environment consisting of moving sources of blue and red light.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 156–164, 2011.

work, we are interested in exploring robotic systems that use similar processes to model their own thinking, and then learn how to change their behavior with little recourse to physical trials or hand-crafted simulators. Understanding self-reflective processes may help make more robust adaptive systems that operate in rapidly changing physical environments, and shed light on these processes in nature [8].

In previous work we have explored how a real robot enhances its adaptation by exploiting self-models of its own morphology [4]. The robot was able to continuously improve its internal models while exploiting propioceptive data collected during its functioning. These models were used to explore new compensatory behaviors. The results suggested that a robot might be more resilient when self-modeling certain source of perturbation, gaining the capabilities of self-representation, prediction, and aided exploration of the source.

Here we wish to take this concept a step further, by having a robot model its own *controller* as well. Just as a robot benefits from modeling its own morphology and then using that model to determine how to best compensate for a new situation, can a robot benefit from modeling its own *controller*, then use it to compensate for a new situations?

The first question to answer is why a robot would need to model its own controller at all, instead of directly accessing and manipulating it. The reasons for this are many fold: First, there are many aspects of a control system that cannot be easily modeled, even if its architecture is perfectly known: Sensor and actuation lag time, noise, and computational errors and delays, for example. Second, the controller may change in unanticipated ways due to failure or change in the environment. Modeling an existing controller also takes time and effort, and direct manipulation of a controller could require an unwarranted increase in software and hardware complexity. Finally, in some cases the controller of a robot is simply inaccessible – either locked by design, or obfuscated by legacy code. The ability to modify performance of an existing controller without directly accessing it also serves as a safe adaption strategy, since the original controller is never modified and therefore its behavior can be restored at any time. This process may also shed light on the evolution of more opaque controllers such as biological nervous systems.

The approach we use here is based on the assumption that there are two controllers; one reflecting on the other, in the same way that metacognition is the ability to reflect upon one's own mental processes and to self-regulate them. Such metacognitive processes are recognized to be present in humans, non-human primates, and a few other mammals [3,10]. It has been recently demonstrated to exist in the rat as well [2], suggesting that it might be applicable to simpler systems such as robots.

The theoretical framework for human metacognition was initially laid out by Nelson and Narens [9]. They proposed that mental activity occurs at a higher *meta* (reflective) level and at a lower *object* (innate) level. The reflective level contains simulations of the object level and interacts by means of two information streams: monitoring and control. Monitoring is the process of observing the processing of the object level and control is the process of modifying the object level. A thought experiment in metacognition was proposed by Minsky [6]. Minsky suggested dividing an artificial brain in two parts. While the input-outputs of the first part (A-brain) are connected to the external world, the second part (B-brain) is only connected to the A-brain; thus A is the only world seen by B (Figure 1a). As proposed by Minsky, the B-brain might help to the A-brain even without having access to the real world, and

by just looking at the activity of the A-brain. Simple questions such as *Are you repeating*? *Are you feeling better?* And *How do you think?* might help to produce a better brain state in the world.

Other studies have examined metacognition in computation [1] with a focus on monitoring a set of variables that are known to be relevant to a specific problem solver or a learning process. An example of this is adjusting learning coefficients, such as weights, during a machine learning task.

We started our investigation in machine self-reflection [11] with the architecture presented in Figure 1b. A simulated wheeled robot (Figure 1c,d) was driven by a pure reactive innate brain. We showed how self-models generated by the reflective brain were useful to produce compensatory resilient behaviors when filtering the innate outputs. In this paper we investigate self-reflection upon time varying causal innate controllers implemented with recurrent neural networks (RNN). We also investigate the effect of filtering the inputs and outputs of the innate controller and the effect of adding time correlated noise to the outputs during monitoring.

The remainder of this paper is as follows: Section 2 describes the innate causal controller. Section 3 describes the process of self-modeling causal controllers using recurrent neural networks. Section 4 presents results on the synthesis of input and output modifiers as well as the effect of adding different amounts of noise to the innate controller outputs. Section 5 contains the conclusions of this investigation.

2 Innate Causal Brain

First, we synthesized an innate robot controller under a highly dynamic environment that contains sources of blue and red light moving in random circular patterns. We evolved the controller such that the robot seeks blue light while avoiding red light. A RNN implemented a controller that mapped the robot sensor inputs to motor outputs, as shown in Figure 2F, referred to as the *Innate-NN*. Four input neurons are fed by the light sensors z^k: $k = \{0...3\}$. The network contains two hidden nodes and two output nodes that generate the left u_0 and right u_1 motor signals. The output y_k of neuron k is computed as

$$y_k = \phi\left(\sum_j w_{kj} x_j - \theta_k\right) \tag{1}$$

where $\phi(\cdot)$ is the sigmoid activation function, x_j are the input signals, w_{kj} are the connection weights and θ_k is the threshold of neuron k. The controller is represented by a genome i of $N_i = 33$ scalar parameters (in the range $[-1, 1]$): 24 connection weights, 8 activation thresholds, and one motor scaling factor α. The reward perceived by a robot is defined in equation (2) by assigning a positive (negative) reward to the amount of blue (red) light intensity that is collected during the evaluation of controller i under environment e after a period of T time steps.

$$F_t^{e,i} = \int_0^t (z_t^{0,e,i} - z_t^{1,e,i} + z_t^{2,e,i} - z_t^{3,e,i})\, dt \tag{2}$$

$$F^i = \prod_{e=1}^{N_E} F_T^{e,i} \tag{3}$$

To avoid exploiting the peculiarities of a unique environment, we used a set of N_e=3 randomly generated environments and we defined the fitness of a candidate

controller as equation (3) indicates. Due to perceptual aliasing[1] an optimal motor action $U_t = \{u_t^0, u_t^1\}$ cannot be purely determined by the sensor state $Z_t = \{z_t^0, \ldots, z_t^3\}$ at a single time t. Thus, in order to provide predictive capabilities to the robot we have decided to use a time-dependent causal controller using a RNN. The algorithm runs with a probability of mutation $p_m = 1/N_i$ using Cauchy mutation and a probability of crossover $p_c = 0.9$. The population size is set to 30 individuals per generation.

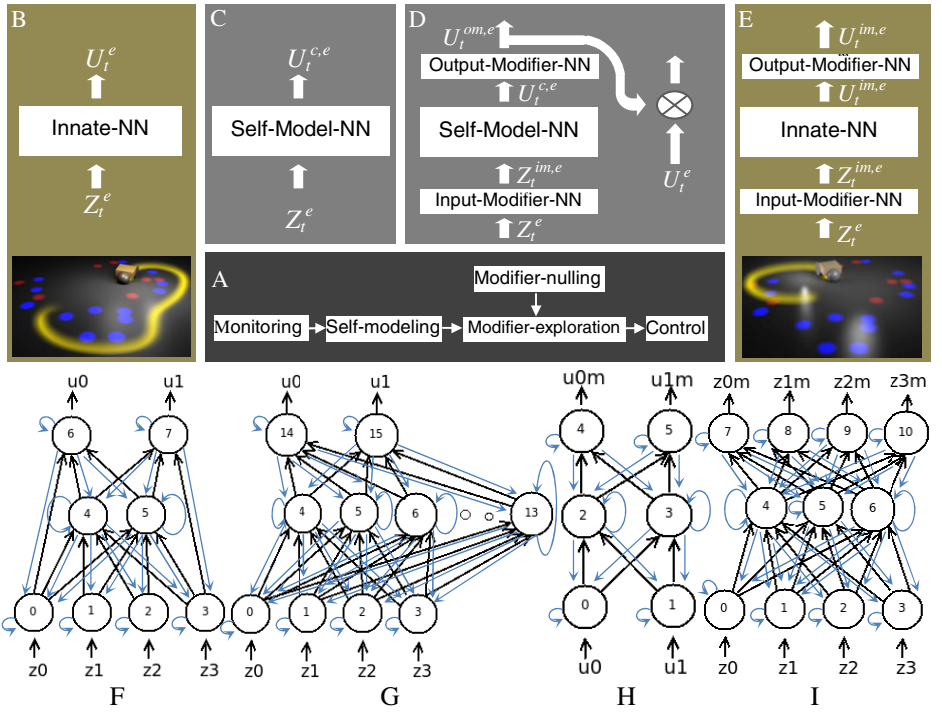


Fig. 2. Synthesis of the self-reflection process. A: The algorithm. B-E: Details of the main processing steps. F: The Innate-NN, four input nodes *{0, 1, 2, 3}* receive measurements z^k: $k = \{0 \ldots 3\}$, the network contains two hidden nodes *{4, 5}* and two output nodes *{6, 7}* generating the motor outputs u^k: $k = \{0, 1\}$. G: Self-model-NN, receives the same inputs and outputs as the innate neural architecture, however, there is a large number of hidden units allowing more complexity. The modifier NN's are in charge of modulating the inputs (I) and outputs (H) that are seen by the Innate-NN during the robot operation.

3 Self-modeling Causal Innate Controllers

We approximate the innate controller with a candidate controller c implemented with a generic RNN controller (Self-model-NN) that has the same number of input and output nodes as the Innate-NN, though a much larger number of hidden units.

[1] Different locations trigger the same sensor state, albeit requiring different control actions.

Figure 2G show this network when considering 10 hidden units, 100 connection weights, 16 activation thresholds and one output scalar factor α.

In this self-modeling stage, let us also introduce an unanticipated environmental change. Suppose that the environment reward suddenly changes: Now the blue light is bad (like poison) and the red light is good. The innate controller is now unoptimal, and fitness reward is decreasing.

We allowed the robot to operate freely under an environment e executing its Innate-NN controller and we collected vector time series of sensor $Z^e = \{Z_t^e : t \in T\}$, motor $U^e = \{U_t^e : t \in T\}$ and reward $F = \{F_t : t \in T\}$. We fed the inputs of each candidate Self-model-NN controller c with the recorded time series of sensor data Z^e, resulting in a predicted motor actuation data $U^{c,e} = \{U_t^{c,e} : t \in T\}$ (Figure 2C). We then measured the quality of each candidate self-model controller c by its ability to reproduce the same input-output patterns as those observed during the operation of the Innate-NN controller on different environments. To find the best self-model, we minimized the distance $D(c)$ described by equation 4.

$$D(c) = \sum_e \int_0^T \left\| U_t^{c,e} - U_t^e \right\| dt \tag{4}$$

Figure 3 shows the convergence of self-models to innate behavior (in red). After 100000 generations self-model controllers are able to drive the robot very similarly as the innate controller does over the training scenario (a,c,d) and even predict the behavior of the innate controller under the test scenario (b,e,f). The similarity of motor commands (c,e) increases when minimizing the distance $D(c)$, resulting robot trajectory becomes closer to the innate trajectory (a,b). Examples of the convergence of resulting sensor channels are shown in (d,f).

4 Synthesis of Input-Output Modifiers

In this section we describe the synthesis of the input-output modifier RNN's that are illustrated in Figure 2H,I. During the controlling stage (Figure 2A) modifiers are in charge of filtering the innate sensor and motor signals. The challenge is how to train these modifier networks in such a way that (i) the reward of the robot goes up again and (ii) by only exploiting past experience, without any new trials.

The method is as follows: Given the time series of sensor Z^e and motor U^e data, we can estimate the quality of candidate modifiers by integrating the observed fitness variation ΔF_t, described in equation 5, *only* in while the modified controller motor action U_t is similar to the already recorded motor action U_t^e that was tested in reality at a given time t. We estimate this similarity by means of the binary function $\gamma_1(U_t)$ defined in equation 7 as function of a threshold $\beta = 0.06$ or by using the continuous function $\gamma_2(U_t)$ defined in equation 8.

Equation 6 shows the normalized fitness related to each candidate modifier. The confidence of the estimation is proportional to the integral across time of $\gamma(U_t)$ which represents either $\gamma_1(U_t)$ or $\gamma_2(U_t)$. Intuitively, the idea is to use the information of fitness variation that was already monitored during the robot operation.

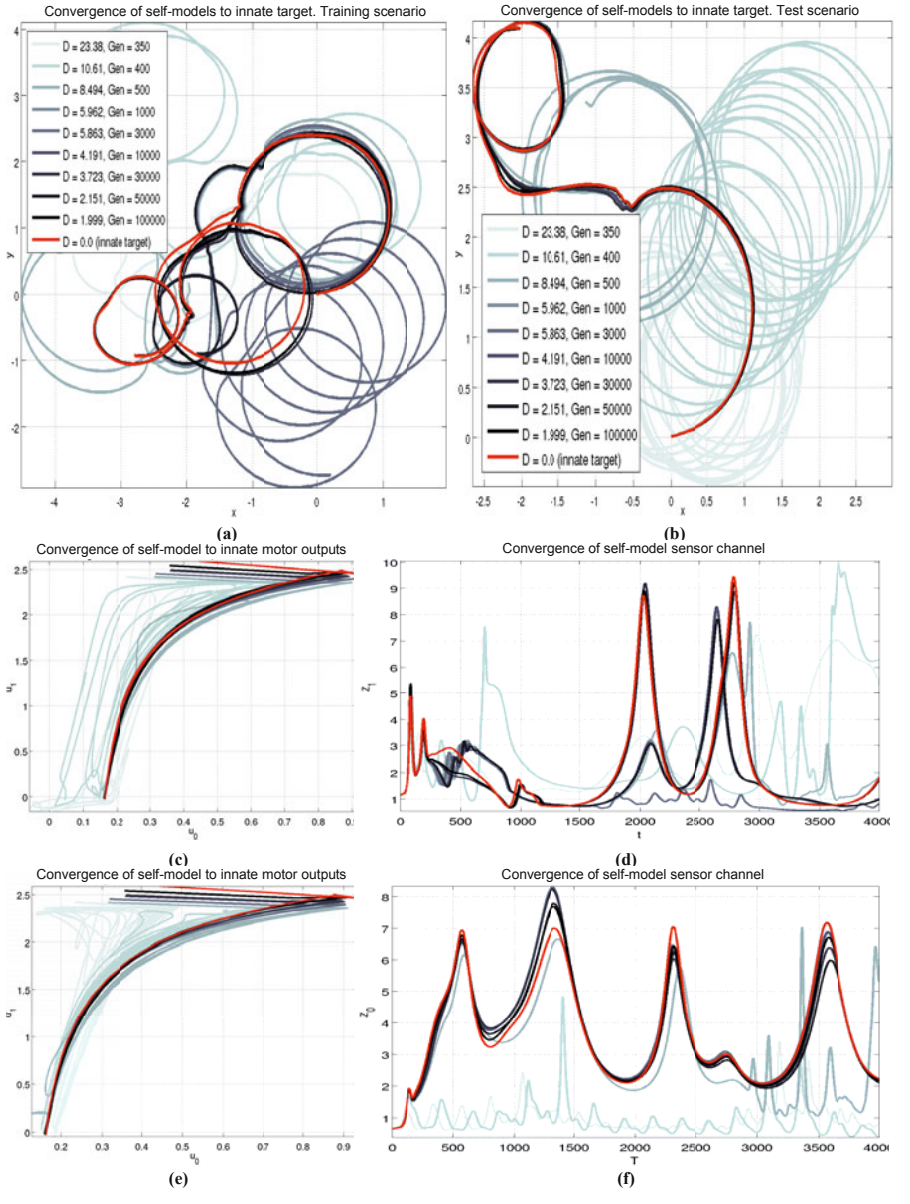


Fig. 3. Convergence of self-models as result from the genetic minimization of *D(c)*. The innate target behavior is shown in red. The self-model behavior is shown under training (a,c,d) and (b,e,f) test scenarios. The similarity of motor commands (c,e) increases when minimizing the distance *D(c)*, resulting robot trajectory becomes closer to the innate trajectory (a,b). Examples of the convergence of resulting sensor channels are shown in (d,f).

$$\Delta F_t = \frac{F_{t+\Delta t} - F_t}{\Delta t} \qquad (5) \qquad \hat{f} = \frac{\int_0^T \gamma(U_t)\Delta F_t dt}{\int_0^T \gamma(U_t) dt} \qquad (6)$$

$$\gamma_1(U_t) = \begin{cases} 1 & if \ \dfrac{\|U_t - U_t^e\|}{\|U_t\| + \|U_t^e\|} \le \beta \\ 0 & otherwise \end{cases} \qquad (7) \qquad \gamma_2(U_t) = 1 - \frac{\|U_t - U_t^e\|}{\|U_t\| + \|U_t^e\|} \qquad (8)$$

Figure 4 below shows activation patterns of $\gamma_1(U_t)$ (a,b) and $\gamma_2(U_t)$ (c) for candidate modifiers explored during self-reflection. The domain axes represent time during evaluation of the innate robot behavior in reality and time during self-reflection. The value of 1.0 (black in c) corresponds to maximum similarity while 0.0 (white in c) represent the highest discrepancy. Time segments showing increasing levels of similarity and high expected fitness are extracted during self-reflection (a). This process is enhanced when adding time correlated noise to the output of innate controllers during fitness monitoring (b). Results indicate that there is no *a priori* correlation between the activations of $\gamma(U_t)$ and the resulting fitness of a candidate.

We initialized the input/output modifiers such that their action is transparent at the beginning of self-reflection. We call this a *nulling* stage that consists on performing

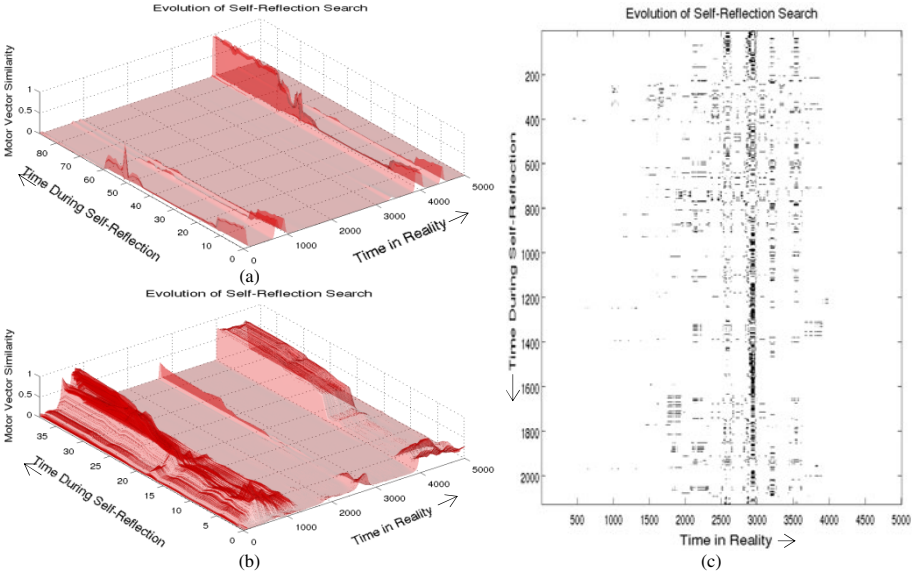


Fig. 4. Activations of $\gamma_1(U_t)$ (a,b) and $\gamma_2(U_t)$ (c). The value of 1.0 (black in c) corresponds to maximum similarity while 0.0 (white in c) represent the highest discrepancy. The effect of adding time correlated noise to the outputs of the innate controller during evaluation in reality is illustrated in (b). Binary activations illustrated in (c) are a good example of how different time structures are extracted during self-reflection.

Table 1. Comparisons of self-reflection versus classical ER techniques. When the environment changes (blue is bad and red is good) the innate controller drives the robot to obtain very low fitness (-644). Recovery using self-reflection allows achieving a fitness of 56 after four hardware trials. A traditional innate recovery would have taken 40.000 trials to achieve similar results.

Design Option	Num. of physical trials	Resulting fitness
Output Modifier, self-ref.	*4 trials.*	56.2 ± 40
Output Modifier, classic ER.	*2000 trials.*	30.4 ± 55
Input Modifier, self-ref.	*4 trials.*	35.8 ± 66
Input + Output Modifier, self-ref.	*10 trials.*	60.3 ± 123
Innate formation.	*150.000 trials.*	-644.3 ± 52
Innate recovery, classical ER.	*40.000 trials.*	73.7 ± 80

genetic search over the modifier search space until their output accurately mimic corresponding inputs. The solutions are then used as seeds for the actual modifier synthesis search process.

5 Conclusions

We conclude that: (1) Accurate self-models of RNN innate controllers can be obtained using the proposed method. (2) Resulting self-models allow prediction of the robot innate behavior even under environments whose dynamics differs from the training scenario. (3) Self-reflection allow our robot to recover quickly; investing 4 or 10 hardware trials to achieve a level of fitness that otherwise would have taken thousands of trials with traditional ER techniques. (4) There is an advantage of inducing motor noise while monitoring the innate robot behavior. (5) We notice the need of pre-calibration of the modifiers prior self-reflection. The large amount of hardware trials is one of the main limitations of ER. Self-reflection appears as a promising alternative to expand the domain of applications of this field.

Acknowledgements

This work has been supported in part by the U.S. National Science Foundation (NSF) Creative-IT program, grant #0757478 and Chilean research FONDECYT project number #3080048.

References

1. Cox, M.T.: Metacognition in Computation: A selected research review. Artificial Intelligence 169(2), 104–141 (2005)
2. Foote, A., Crystal, J.: Metacognition in the Rat. Current Biology 17(6), 551–555 (2007)
3. Hampton, R.: Rhesus monkeys know when they remember. Proceedings of the National Academy of Sciences 98(9), 5359 (2001)

4. Bongard, J.C., Zykov, V., Lipson, H.: Resilient Machines through Continuous Self-Modeling. Science 314(5802), 1118–1121 (2006)
5. Brooks, R.A.: Intelligence Without Representation. Artificial Intelligence 47(1-3), 139–159 (1991)
6. Minsky, M.: The Society of Mind. Simon & Chuster Inc., New York (1986)
7. Minsky, M.: Interior grounding, reflection, and self-consciousness. In: Proceedings of International Conference on Brain, Mind and Society. Tohoku University, Japan (2005)
8. Minsky, M.: The Emotion Machine. Simon & Schuster Inc., New York (2007)
9. Nelson, T., Narens, L.: Metamemory: A theoretical framework and new findings. The Psychology of Learning and Motivation 26, 125–141 (1990)
10. Smith, J., Shields, W., Washburn, D.: The comparative psychology of uncertainty monitoring and metacognition. Behavioral and Brain Sciences 26(03), 317–339 (2004)
11. Zagal, J.C., Lipson, H.: Self-Reflection in Evolutionary Robotics: Resilient Adaptation with a Minimum of Physical Exploration. In: Proceedings of the Genetic and Evolutionary Computation Conference, Late Breaking Papers, GECCO (2009)

Swarm-Bots to the Rescue

Rehan O'Grady[1], Carlo Pinciroli[1], Roderich Groß[2], Anders Lyhne Christensen[3], Francesco Mondada[2], Michael Bonani[2], and Marco Dorigo[1]

[1] IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
[2] LSRO, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[3] DCTI, Lisbon University Institute, Lisbon, Portugal

Abstract. We explore the problem of resource allocation in a system made up of autonomous agents that can either carry out tasks individually or, when necessary, cooperate by forming physical connections with each other. We consider a group transport scenario that involves transporting broken robots to a repair zone. Some broken robots can be transported by an individual 'rescue' robot, whereas other broken robots are heavier and therefore require the rescue robots to self-assemble into a larger and stronger composite entity. We present a distributed controller that solves this task while efficiently allocating resources. We conduct a series of real-world experiments to show that our system can i) transport separate broken robots in parallel, ii) trigger self-assembly into composite entities when necessary to overcome the physical limitations of individual agents, iii) efficiently allocate resources and iv) resolve deadlock situations.

Keywords: self-assembly, task allocation, swarm intelligence, search and rescue, cooperation, group transport, autonomous robots, multi-robot systems, swarm robotics.

1 Introduction

Self-assembling robotic systems are made up of autonomous agents that are able to physically connect to one another to form larger composite entities. Self-assembling systems in which the individual agents are themselves simple independent mobile robots potentially share the benefits both of distributed multi-agent robotic systems and of more conventional monolithic robots. In common with distributed multi-agent systems, such self-assembling systems are well suited to parallel task execution, and tend to be relatively inexpensive, robust and scalable. However, by connecting to each other and working together, such systems can also overcome the individual physical limitations of the system's constituent agents and thus carry out tasks that in the past might have required a larger traditional monolithic robot.

However, there is an intrinsic problem of resource allocation that must be solved before self-assembling systems can achieve this kind of flexibility and realise their potential. In particular, a self-assembling system must be able to

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 165–172, 2011.

determine when parallel or collective behaviour is more appropriate, and then be able to distribute resources to reflect this analysis. This problem of resource allocation has been largely ignored in the self-assembly literature. There is a large body of literature on task allocation in non-self-assembling multi-robot systems. However, little of this work is directly applicable to self-assembling systems, where the parameters of the task must determine the nature and extent of physical cooperation at the expense of parallel execution.

In this paper, we explore the problem of resource allocation using a real world group transport scenario. In our scenario, dedicated 'rescue' robots must find broken, immobile robots and transport them to a designated repair zone. The broken down robots can either be single agents or pre-assembled composite robotic entities. A single broken down agent can be transported by a single rescuing agent, whereas a broken down composite entity is sufficiently heavy that multiple rescue robots must self-assemble in order to effect the rescue. The system has no a priori knowledge either of the number of rescue robots or of the number and size of the broken entities.

We present a distributed controller that solves the above task while efficiently allocating resources. Each rescue robot tries to move any broken robots that it finds, and independently determines whether or not the object is successfully being moved. Based on this determination, the rescuing robot uses local communication to either attract or repel other rescue robots. We present a series of experiments with real robots using our controller. The contributions of this study are as follows. Firstly, we demonstrate a new use of self-assembly as a response mechanism. Secondly, we demonstrate a distributed task allocation mechanism based on local attraction and repulsion that is applicable to groups of mobile self-assembling agents. Finally, we demonstrate a group transport control mechanism that improves on previous implementations in both efficiency and flexibility.

2 Related Work

Kube and Zhang [5,6] conducted a series of experiments in which a group of physical robots was transporting a heavy object. They observed that the robots could end up pushing the same object in opposing directions. As a result of this, the transport object could become stuck. To resolve this problem, they integrated into their control policy a recovery mechanism that was inspired by the group transport of the ant species *Pheidole crassinoda* [9]. In this species, ants were reported both to undergo changes in their group transport arrangements and to recruit nest mates if an object resisted motion (Kube and Zhang's robots mimicked the group transport rearrangement behaviour).

In simulation, Perez-Uribe *et al.* [8] investigated a system in which a group of robots is required to find and transport multiple objects of two sizes: small and large. The allocation of robots to the objects was irreversible, thus creating a deadlock potential in the case of an immovable object.

Ijspeert *et al.* [4] studied a system of physical robots in which two physically cooperating robots could pull long sticks out of the ground (the sticks could not be removed by a single robot). When a single robot tried to remove a stick, it would wait for another robot to arrive. The optimal waiting time was computed as a function of the environmental parameters.

Groß and Dorigo [1,2,3] used computer simulations to study the transport of heavy objects by groups of self-assembling robots. The control policies were designed by evolutionary algorithms. In [2,3], it was assumed that only a single transportable object was present. The system in [1] could in principle cope with multiple objects, however, each robot would always attempt to transport the closest object within its perceptual field. The allocation of robots was irreversible and did not depend on the objects' resistance to motion.

Tuci *et al.* [10] conducted an experiment with physical robots that could self-assemble and transport a heavy object. The transporters were programmed to suspend their transport whenever they perceived an unconnected robot (allowing the latter to join the group). Thus, if a robot permanently failed to join the pulling structure, a deadlock occurred.

3 Platform and Experimental Setup

The platform we use is the swarm-bot robotic platform [7]. The swarm-bots platform consist of a number of autonomous robots called *s-bots*, see Fig. 1 (left). The s-bot is 12 cm high without its camera turret, and has a diameter of 12 cm. Each s-bot is equipped with a gripper that enables physical connections between s-bots. The turret holding the gripper can rotate independently of the chassis, that contains a differential drive system composed of combined tracks and wheels. This allows an s-bot to exert force in any direction when grasping another s-bot or object.

Each s-bot is equipped with torque sensors on its track motors that let the robot determine whether it is successfully moving a gripped object that it is

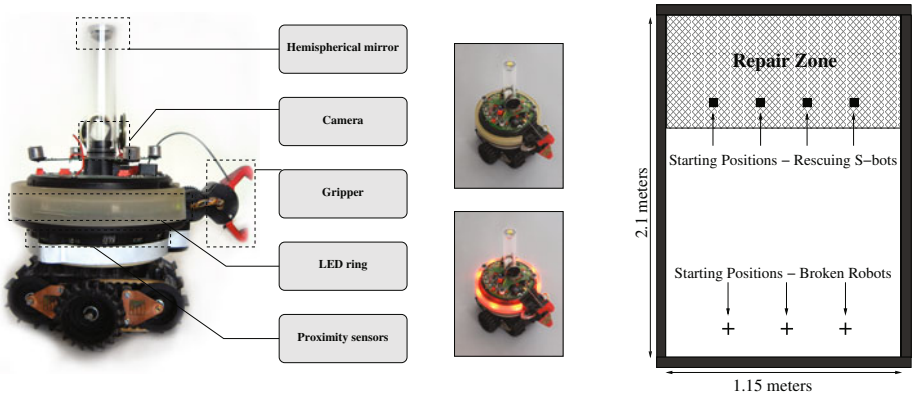


Fig. 1. Left: The s-bot. Centre: S-bot LEDs off and on. Right: The arena.

trying to transport. If the s-bot is moving the object, the s-bot's tracks will be rotating and the motor torque will be low. If, however, the object is not moving (for example, because it is too heavy) the rescue robot's tracks will be blocked and the torque will be high.

The s-bot camera records panoramic images reflected in a hemispherical mirror mounted above the s-bot chassis in a transparent perspex tube. Each s-bot has a semi-transparent ring housing eight sets of RGB coloured LEDs. Depending on light conditions, the camera can detect illuminated LEDs on other s-bots up to 50 cm away. A strong external light source can be seen up to 4 m away.

Our experimental setup requires 'rescue' s-bots to search the arena shown in Fig. 1 (right), to find broken robots and then to transport them to the designated repair zone (darker floor). A light source is located two meters outside of the arena beyond the repair zone (not shown in Fig. 1). Transporting s-bots perform phototaxis to ensure that they transport the broken robots in the correct direction towards the repair zone.

All of our experiments involve either one or two rescuing robots and either one or two broken robots. We use two types of broken robot. We refer to a single broken s-bot as a *1-s-bot broken robot* (a single rescuing s-bot is able to transport a 1-s-bot broken robot). We refer to a broken robot that is a composite entity made up of two physically connected s-bots as a *2-s-bot broken robot* (the rescuing s-bots must team up in order to collectively transport a 2-s-bot broken robot). We consider broken robots to be immobile. However, we make the simplifying assumption that broken robots are still able to use their LEDs to signal that they require assistance.

4 Controller

The desired behaviour of our multi-agent system is for the rescue s-bots to locate and transport the broken robots to the repair zone. Ideally, based on the properties of the broken robots (size, weight), only the minimum required number of rescue s-bots should be allocated to transport each broken robot, thus leaving as many rescue s-bots as possible idle or available for other tasks.

We designed our distributed controller to use only local sensing and communication. The control logic executed independently by each rescue s-bot is shown in Fig. 2. A rescue s-bot searches for any broken robots by performing a random walk with its blue LEDs illuminated (`Random Walk`). Broken robots signal that they need help by illuminating their red LEDs. When a rescue s-bot detects red LEDs without any nearby green LEDs, it assumes that it has seen a broken robot that is not currently being rescued and heads towards the red LEDs (`Goto Red Entity`). If more than one red entity is seen, the rescuing s-bot picks one of the red entities at random. It then grips the broken robot (`Grip Red Entity`) and tries to pull the broken robot to the repair zone by heading towards the light source (`Pull Towards Light`) with its green LEDs illuminated.

As long as a rescue s-bot is successfully pulling a broken robot (low track torque), it stays green. This tells other rescue s-bots within camera range not

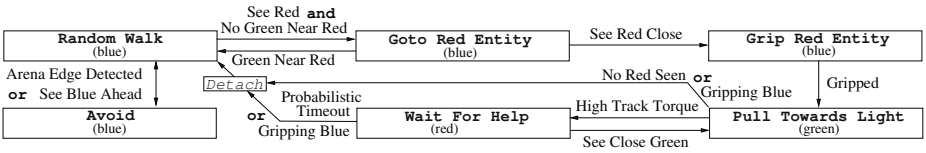


Fig. 2. Control logic that each rescue s-bot executes independently

to go towards the broken robot it is pulling. However, if the rescue s-bot fails to move the broken robot (high track torque measured consistently for 5 seconds), the rescue s-bot stops pulling and illuminates its red LEDs (`Wait For Help`). At this point, the rescue s-bot is indistinguishable from the broken robot that it is still gripping. Thus, other rescue s-bots will approach and grip either the broken robot or any rescue s-bots that are attached to the broken robot and are in state `Wait For Help`. When a new rescue s-bot attaches, it will turn green as it attempts to pull the broken robot. Any attached s-bots in state `Wait For Help` are prompted by the sight of green LEDs to try to pull again. If the new larger number of attached rescue s-bots is now sufficient to move the broken robots, all of the rescue s-bots will stay green and thus prevent any further rescue s-bots from approaching. Otherwise, after 5 seconds of high torque, they will realise that the broken robot is still not movable, and will switch to state `Wait For Help`. This process repeats itself, with progressively more rescue s-bots attaching to the broken robot until there are enough rescue s-bots to move the broken robot. In this way, the system automatically finds the minimum group size capable of moving a broken robot.

If there are several large composite broken robots present, and a small number of rescue s-bots, it is possible that the individual rescue s-bots might randomly distribute themselves among the broken robots in such a way that none of the large broken robots can be moved. To prevent this type of deadlock situation, a rescue s-bot in state `Wait For Help` has a low probability of detaching. A detached robot returns to state `Random Walk` and thus turns blue, which signals to any s-bot that might be gripping the detached robot that it should in turn detach. Rescuing robots also detach if they no longer detect any red LEDs—in our experiments, the broken robots detect arrival in the repair zone using ground sensors, and switch off their red LEDs. Together, these two detachment mechanisms ensure that as long as there are enough rescue s-bots present in the environment to move any given broken robot, the rescue s-bots will eventually combine forces to move the broken robot.

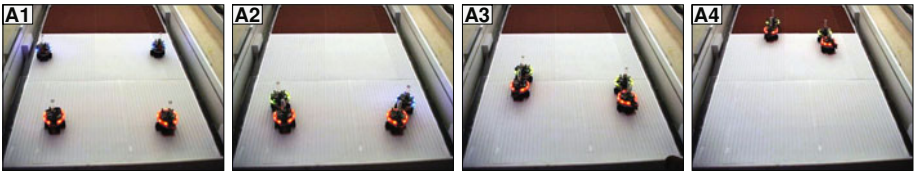
5 Results

We carried out four sets of real-world experiments. Each set of experiments is defined by the number of rescuing s-bots, the number of broken robots and the size of the broken robotic entities. The rescuing robots have no a priori knowledge

of the experiment configuration. Videos of the experiments can be found on the web at: http://iridia.ulb.ac.be/supp/IridiaSupp2009-011.

Rescuing Broken Robots in Parallel

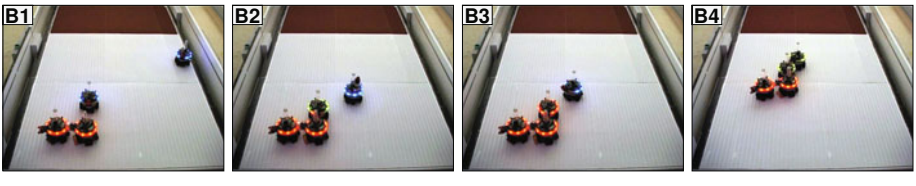
We conducted 5 trials of an experiment with two rescue s-bots and two 1-s-bot broken robots (photos A1-A4 are of a single trial). In each experiment, the system successfully allocated a single rescue robot to each of the broken robots and the broken robots were transported in parallel to the repair zone.



This experiment shows that when possible the system correctly 'chooses' parallel execution (the incorrect choice would be for two rescue s-bots to assemble to the same broken robot).

Physical Cooperation to Rescue a Broken Robot

We conducted 5 trials of an experiment with two rescue s-bots and a single 2-s-bot broken robot (photos B1-B4 are of a single trial). In each trial, one rescue s-bot found the broken robot, then tried and failed to move the broken robot alone. The attached rescue s-bot waited for help, and after the other rescue s-bot attached, together the two rescue s-bots succeeded in transporting the broken robot to the repair zone.

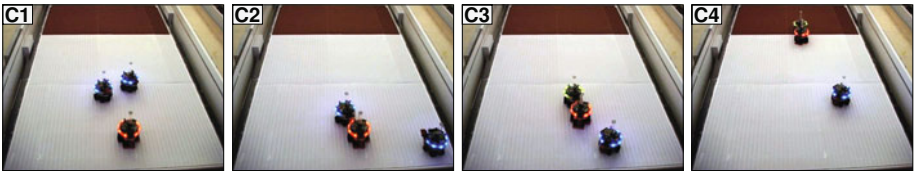


In one of the trials, the initial attachment configuration of the rescue s-bots failed to move the broken robot. One of the rescue s-bots detached based on the probabilistic time out twice, but reattached both times. In the final configuration the rescue s-bots succeeded in transporting the broken robot to the repair zone.

This experiment shows that the system correctly allows the rescue s-bots to physically coordinate in order to solve a task that is beyond the physical capacities of a single s-bot.

Efficiency Gains Through Group Size Regulation

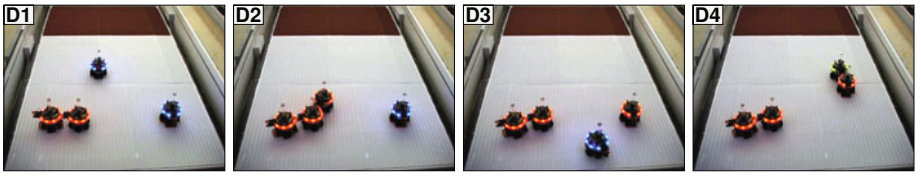
We conducted 5 trials of an experiment with two rescue s-bots and a single 1-s-bot broken robot (photos C1-C4 are of a single trial). In each trial, a single rescue s-bot connected to the broken robot, and successfully transported it to the repair zone. The other s-bot continued to explore the arena without attempting to attach to the moving broken robot.



This experiment shows that the system correctly allocates the minimum number of rescue s-bots (in this case 1 s-bot) to a task, and leaves other rescue s-bots free to potentially carry out other tasks.

Reallocation of Resources in a Deadlock Situation

We conducted 5 trials of an experiment with a single rescue s-bot and two broken robots: one 1-s-bot broken robot and one 2-s-bot broken robot (photos D1-D4 are of a single trial). In this experiment, we initially 'break' the 2-s-bot broken robot (i.e., illuminate its red LEDs). We allow the rescuing s-bot to find and attach to the heavy broken robot before we 'break' the 1-s-bot broken robot.



In four out of the five trials, the rescue s-bot probabilistically timed out, explored the arena, found and attached to the 1-s-bot broken robot and successfully transported it to the repair zone (in one of these successful trials the rescue s-bot first re-attached to the 2-s-bot broken robot and timed out again). In a single trial, the rescue s-bot failed to detach correctly due to a hardware failure.

This experiment shows that the system correctly resolves a deadlock situation where a rescue s-bot is attempting an impossible task, and manages to reallocate resources (the rescue s-bot) to another task that is feasible.

6 Conclusion

In this study, we presented a distributed controller that for the first time tackles the problem of resource allocation in a self-assembling robotic system. We conducted real world experiments in a group transport rescue scenario which showed that our distributed control logic displays several important properties. In particular, our system was able to maximise parallel execution (and hence efficiency) by allocating the minimum number of agents required to solve tasks of varying magnitudes. In addition, the system included a reset mechanism which allowed it to resolve potential deadlock situations (when all agents are distributed among tasks in a sufficiently sparse way that none of the tasks are solvable). We discovered that this mechanism also allowed for potentially beneficial random reconfiguration of spatial arrangements within a single transport rescue group.

We are currently working on testing the scalability of the system with larger numbers of robots in simulation, and on abstracting the fundamental dynamics of our system so as to apply them to other self-assembly scenarios.

Acknowledgements

This work was supported by the Sixth Framework Programme of the European Commission in the form of the IST FET project SWARMANOID (contract IST-022888), of the Marie Curie Early Stage Research Training Site COMP2SYS (contract MEST-CT-2004-505079), and of a Marie Curie Intra-European Fellowship (contract MEIF-CT-2006-040312). Marco Dorigo acknowledges support from the F.R.S.–FNRS of the Belgian French Community, of which he is a research director, via the VIRTUAL SWARMANOID project. The information provided is the sole responsibility of the authors and does not reflect the European Commission's opinion. The European Commission is not responsible for any use that might be made of data appearing in this publication.

References

1. Groß, R., Dorigo, M.: Group transport of an object to a target that only some group members sense. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 852–861. Springer, Heidelberg (2004)
2. Groß, R., Dorigo, M.: Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling. Adapt. Behav. 16(5), 285–305 (2008)
3. Groß, R., Dorigo, M.: Towards group transport by swarms of robots. Int. J. of Bio-Inspired Computation 1(1-2), 1–13 (2009)
4. Ijspeert, A.J., Martinoli, A., Billard, A., Gambardella, L.M.: Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment. Autonomous Robots 11(2), 149–171 (2001)
5. Kube, C.R., Zhang, H.: Stagnation recovery behaviours for collective robotics. In: Proc. of the 1994 IEEE/RSJ/GI Int. Conf. on Intell. Rob. and Sys., vol. 3, pp. 1883–1890. IEEE Computer Society Press, Los Alamitos (1994)
6. Kube, C.R., Zhang, H.: Task modelling in collective robotics. Autonomous Robots 4(1), 53–72 (1997)
7. Mondada, F., Pettinaro, G.C., Guignard, A., Kwee, I.W., Floreano, D., Deneubourg, J.-L., Nolfi, S., Gambardella, L.M., Dorigo, M.: SWARM-BOT: A new distributed robotic concept. Auton. Robots 17(2-3), 193–221 (2004)
8. Pérez-Uribe, A., Floreano, D., Keller, L.: Effects of group composition and level of selection in the evolution of cooperation in artificial ants. In: Banzhaf, W., Ziegler, J., Christaller, T., Dittrich, P., Kim, J.T. (eds.) ECAL 2003. LNCS (LNAI), vol. 2801, pp. 128–137. Springer, Heidelberg (2003)
9. Sudd, J.H.: The transport of prey by an ant. Pheidole Crassinoda Em. Behaviour 16(3-4), 295–308 (1960)
10. Tuci, E., Groß, R., Trianni, V., Mondada, F., Bonani, M., Dorigo, M.: Cooperation through self-assembly in multi-robot systems. ACM Trans. Auton. Adapt. Syst. 1(2), 115–150 (2006)

Towards an Autonomous Evolution of Non-biological Physical Organisms

Roderich Groß, Stéphane Magnenat, Lorenz Küchler, Vasili Massaras, Michael Bonani, and Francesco Mondada

Ecole Polytechnique Fédérale de Lausanne, LSRO, Station 9,
CH-1015 Lausanne, Switzerland
roderich.gross@ieee.org

Abstract. We propose an experimental study where simplistic organisms rise from inanimate matter and evolve solely through physical interactions. These organisms are composed of three types of macroscopic building blocks floating in an agitated medium. The dynamism of the medium allows the blocks to physically bind with and disband from each other. This results in the emergence of organisms and their reproduction. The process is governed solely by the building blocks' local interactions in the absence of any blueprint or central command. We demonstrate the feasibility of our approach by realistic computer simulations and a hardware prototype. Our results suggest that an autonomous evolution of non-biological organisms can be realized in human-designed environments and, potentially, in natural environments as well.

Keywords: Adaptation, artificial life, evolution, evolutionary robotics, morphology, origin of life, self-assembly, self-organization, self-replication.

1 Introduction

In this paper, we argue that *artificial* evolution of living organisms could, or should, take place in worlds that obey the laws of physics, and where possible, in the natural world. If this were the case, the evolutionary processes would not only be validated but could explore deeply the world's own dynamics and its nonlinear nature [1]. A step towards this direction was made by Floreano and Mondada [2], who proposed an approach to evolve—without human intervention—the brain of a robot that interacts with its physical world. The brain, an artificial neural network, was modeled in software, but as Thompson [3] showed, it could be embedded into an electronic circuit as well. Sims [4,5] investigated computer simulations to evolve both body and brain of organisms (see also [6]). Funes and Pollack [7] investigated computer simulations to evolve static support structures made of realistic components (LEGO bricks), which allowed them to build and test the best solutions in reality. Lipson and Pollack [8] extended this approach by an automatic procedure to manufacture the solutions, in this case, robotic lifeforms.

Different from natural evolution, the aforementioned approaches to artificial evolution are not to the extent self-organized as we would like them to be. For

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 173–180, 2011.

example, they all share a central computer algorithm that decides whether organisms might reproduce or not. Moreover, they make use of dedicated computer algorithms that can produce new organisms on demand, for example, by recombining or varying existing solutions. By contrast, natural evolution is an autonomous, decentralized, and self-organized process that is fully embedded into the physical world. To the best of our knowledge, up to now the study of self-organized evolutionary processes has considered only fairly abstract models, where the genotypes (and genetic operators) are either software entities lacking embodiment [9,10,11,12,13,14] or entities having a rudimentary embodiment only [15].

Our work builds on recent advances in systems capable of macroscopic self-assembly [16,17]. In these systems, as the result of a self-organized process, a set of centimeter-sized building blocks can form composite entities. Several macroscopic self-assembly systems—ranging from purely mechanical parts to fully autonomous robots—proved capable of replicating connected composite entities [18,19,20,21,22]. However, these composite entities did not undergo change, and thus could not evolve (but see [23]). By contrast, we investigate a self-assembling system that is capable of producing a population of embodied organisms (i) which are subject to change through artificial evolution, and (ii) which respond to stimuli in their environment. The defining characteristics of our system are:

1. it is composed of pre-existing building blocks: energy modules, interaction modules, and boundary modules;
2. the modules (and composite entities) float passively in an agitated medium;
3. the energy and interaction modules self-assemble into composite entities;
4. the energy modules transform and store energy provided by the environment;
5. the interaction modules respond to stimuli in their environment;
6. the boundary modules attach to composite entities and thereby form protecting membranes, which inhibit further growth;
7. modules within a same composite entity share their energy;
8. modules without energy are not powered and can thus not actively bind with other modules (however, they can passively bind with active modules);
9. composite entities with an intact membrane replicate by self-assembly;
10. composite entities can break into multiple parts.

In the following we refer to composite entities with an intact membrane as *organisms*. Note that organisms (i) need energy, for example to maintain their connectivity (see item 8), and (ii) can replicate (see item 9). In this study, the organism (phenotype) is identical to the genotype.

The paper is organized as follows. Section 2 describes the simulator that we use to model the physical process. Section 3 explains the process itself. Sections 4 and 5 detail respectively the computational results and a hardware prototype. Section 6 discusses the findings and concludes the paper.

2 Simulation Model

The simulator models the kinematics and dynamics of rigid bodies in two dimensions (2-D) using the open-source Enki simulation toolkit [24]. The 2-D space

is modeled continuously. Time progresses in discrete steps. The environment is a bounded squared world of side length 250 cm. At any moment in time, it is partitioned into distinct regions where light is either present (*day regions*) or not present (*night regions*). The world is populated by physical objects (modules or composite entities). The objects cannot move on their own, but float passively on the ground. Kinetic energy is provided by the flow of air. The flow is composed of two components: a flow in random directions of velocity 280 cm/s and a counter-clockwise circular flow of velocity 160 cm/s around the world's center. The forces exerted by the flow of air result in random or circular motion patterns. In nature, such motion patterns could result from ocean currents, gravitational fields, or Brownian motion. The combination of circular motion patterns with a squared world is expected to provide nonlinearity to the dynamics of the system.

The system's basic building blocks, the modules, are modeled as squares of side length 7 cm and of mass 49 g. The modules can physically connect with each other and thereby form composite entities. Each module controls the connectivity of each of its four sides by activating or deactivating it. If the sides of two separate modules are well aligned with each other, a connection is established provided that at least one of the two sides is activated.

The system has three types of modules:

1. The energy module, or *e*-module, harvests, stores, and provides energy. The energy consumption is 1 unit/s for modules of all types. When part of a same composite entity, *e*-modules share instantaneously their energy with all other modules via a power line. They also balance their energy storage over time.
2. The interaction module, or *i*-module, allows composite entities to respond to stimuli in their environment. In this study, the *i*-module can (i) adjust the friction (coefficient) it has by contact with the ground and (ii) perceive whether it is located in a day or night region. The module's behavior is hard-wired as follows: the friction coefficient is 0.2 in day regions, and it is 0.02 in night regions or whenever the module is powered off. The *i*-module is powered on whenever it receives energy through its power lines.
3. The boundary module, or *b*-module, allows composite entities to be encapsulated by a protecting membrane (boundary). Once a *b*-module has attached, the composite entity is prevented from further growth. The *b*-module is powered on whenever it receives energy through its power lines.

3 Origin of Organisms, Replication, and Variation

At the beginning of a trial, the modules are randomly distributed in the world (see Fig. 1a). When an *e*-module retrieves energy from the environment, it gets automatically powered on. It then activates all four of its connection sides. Recall that a connection between two modules can be established only if at least one of the interacting connection sides is active. As the *i*- and *b*- modules do not have energy on their own, at this stage any growth is seeded by at least one *e*-module. When a separate *e*- or *i*- module connects with another module during the

Fig. 1. Left: (a–d): illustration of the growth and replication process. Center/right: generic hardware prototype capable of simulating all aspects of the *e*-, *i*-, and *b*-modules. Center: squared base with four connection sides. The inset shows a hatch in the bottom plate, which controls the mobility of the module, which floats on an air table. Right: fully assembled prototype (battery removed).

growth phase, it deactivates its two lateral connection sides. As a consequence, the modules form polymers (i.e., linear chains) of arbitrary length. Once a *b*-module connects to either end of such a polymer, a signal propagates to the other end, and thereby a membrane is established (see Fig. 1b). The membrane prevents further extensions on either side of the polymer during its lifetime. The self-assembly process thus results in linear organisms that are composed of ≥ 1 *e*-modules, ≥ 0 *i*-modules, and 1 *b*-module.

Organisms attempt to *replicate* at any moment in time. To do so, the *b*-module activates one lateral connection side. Once a module of correct type has attached to this side, the *b*-module sends a signal to the next module in the organism chain. The replication process then proceeds by copying elements, one by one, similar to the Watson-Crick base pairing. In our case, modules pair only with modules of the same type (see Fig. 1c). Mismatches in type are recognized and the modules released. Once the replication has completed, the two organisms split apart (see Fig. 1d).

In some situations composite entities (including organisms) break apart. First, this happens when a composite entity has no energy left (i.e., *selection* occurs via the environment). Second, this can happen when a module of a composite entity detects a local inconsistency. For example, this is the case when two *b*-modules connect at about the same time to both ends of a polymer. Third, composite entities can break apart when experiencing a high impulse during collision. The composite entity then splits into two or more parts. The aforementioned situations can lead to *variations* of the organisms (and composite entities). The resulting composite entities can form new polymer structures (if lacking a membrane), for example, by recombining with each other.

The exact logic governing the local interactions during growth and self-replication is coded in the form of finite state machines (FSM). The entire process is regulated by local information only. Communication between two adjacent modules is limited to a single byte per time step in each direction (regardless of the size of the composite entity).

4 Results

To assess our system, we put 150 modules—50 per type—at random positions in the world. At the beginning, the energy storage of each *e*-module was empty. Its capacity was limited to 300 units. When powered, modules of all types consumed 1 unit/s. Energy was provided at a rate of 1.41 units/s by a single day region covering the entire world. After 36000 s (10 h) had elapsed, energy was provided at a rate of 1.90 units/s by three non-overlapping day regions covering each a 7/36th circular segment of the entire world. The day regions were separated by equally-sized night regions (i.e., 5/36th circular segments of the world), which provided no energy. The day and night regions moved at a constant speed similar to the sun relative to the Earth. If a module could remain motionless it would experience a "sunrise" every 1200 s followed by a "sunset" 700 s thereafter. The circular air flow, which was driving the modules, was exactly opposing the circular movement of the day and night regions.

Figure 2 (left) shows a snapshot taken from an experiment at time 2700 s. In general, almost all of the organisms that emerged were not capable of harvesting enough energy to stay alive; typically they died shortly after becoming alive or when reproducing. In the first phase (10 h), the energy was uniformly distributed in the world. Consequently, mobility was not relevant for energy retrieval. This certainly explains the lack of *i*-modules (which control ground friction as a response to light) in the organisms that evolved at this stage [see Fig. 2 (center)]. We repeated the phase 1 evolution ten times and in all cases a few organisms of adequate structure emerged spontaneously and then replicated rapidly until the initial supply of modules was exhausted. In the evolutionary run shown in Fig. 2 (center/right), the population converged to a single species represented by eight identical individuals (seven of which were generated by self-replication). Each individual consisted of 4 *e*-modules and 1 *b*-module. The remaining 18 *e*-modules were attached to these organisms in the form of base pairs.

In the second phase, which started when 36000 s (10 h) had elapsed, energy was not uniformly distributed. At the beginning of phase 2, we observed the

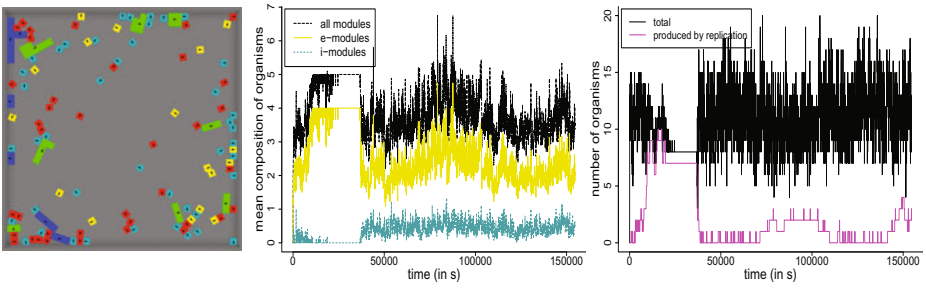


Fig. 2. Left: snapshot taken in a simulation trial (green: organisms, some of which are replicating, blue: polymers without membrane, yellow/turquoise/red: *e*/*i*/*b*-modules). Center: mean composition (i.e., modular makeup) of organisms over time. Right: total number of organisms and number of those produced by replication over time.

extinction of the aforementioned species [Fig. 2 (right)]. During phase 2 new organisms emerged. The most successful ones comprised 6 e-modules, 1 i-module, and 1 b-module. The i-modules enabled the organisms to increase the relative time spent within the day regions. Preliminary analysis suggests that the position of the i-module within the organism was also a crucial factor for survival. As can be seen in Fig. 2 (right), some of the organisms in phase 2 were produced by self-replication. However, different from phase 1 these organisms did not spread in the entire population. A possible explanation for this is that the mean density of energy had dropped from 1.41 units/s in phase 1 to $1.108\bar{3}$ units/s in phase 2. Recall that 1 unit/s is consumed already by the e-module itself. In addition, the environment in phase 2 was highly unpredictable as energy was supplied only in certain regions, which changed over time. The limited size of these regions certainly created competition between the organisms, which—due to their embodiment—could not occupy the same positions.

5 Hardware Implementation

We have designed and built a generic hardware prototype, which can simulate all aspects of the e-, i-, and b-modules. The prototype is shown in Fig. 1 (center/right). It has a size of 7 cm times 7 cm, a total weight of 59 g, and can float on an air table. The module's base [see Fig. 1 (center)] was fabricated using a 3D printer. Its slanted edges facilitate self-alignment when colliding with other modules. A module can attach to other modules on each of its four sides. The connection mechanism is similar to the one reported by Klavins' group in [25]. Each side has two permanent magnets. One magnet is fixed in position with the north pole pointing outwards. The other magnet can be rotated by means of a servomotor, which gives basic control on the level of attraction or repulsion. A hatch in the base can be opened or closed in order to allow the module to immobilize itself: when the hatch is opened, air from the table flows through the opening and as a consequence the module's ground friction increases. The hatch is actuated by a fifth servomotor, which lies flat on the base. The module contains a printed circuit board with a dsPIC33F microcontroller. For inter-module communication, the module has four infrared transmitters. A fifth transmitter (light sensor) is mounted on top of the module, pointing upwards. A 350 mAh lithium polymer battery provides energy.

6 Discussion

In this paper, we proposed a self-assembling system that can allow non-biological evolutionary processes to take place in the physical world. The evolutionary process is fully autonomous, decentralized, and self-organized. It is governed by the organisms' physical interactions with each other and with their environment. The physical interactions are in turn determined by a number of factors. For example, the motion of an organism (which is made possible by the flow of air in the environment) is affected by the organism's mass, center of mass, moment of

inertia, the orientation-dependent surface area, geometry, and the non-uniform friction—all parameters that vary with the organism's modular makeup.

First results obtained in computer simulations are very promising and indicate the feasibility of such evolutionary process. In particular, the system proved capable of generating a population of organisms that had qualities of living beings (e.g., response to stimuli) and that were well adapted to their environment, even when the latter was subject to gradual or sudden changes. Clearly, most of the system's behavior is yet to be explored. For example, we suppose that the role of variations were very limited in the experiments we reported (only a few organisms broke apart and recombined to new solutions). However, we expect the role of variation to become more important when the environment complexity and the organisms' sizes further increase—under these circumstances it should be more difficult to assemble an appropriate solution from scratch.

We have constructed a hardware prototype that implements the required key functionalities apart from energy sharing and energy retrieval. In principle, these functionalities can be simulated in a physical setting by using the prototype's onboard battery and light sensor. As onboard batteries will limit the time of operation, we plan to equip the modules with solar panels and energy sharing facilities [26]. The system should then be capable of exhibiting an autonomous physical evolution in a human designed environment. The ultimate goal would be to design a system that can evolve physical organisms in natural environments.

Acknowledgment

The authors thank Prof. Jarle Breivik for stimulating discussions that helped in the preparation of the manuscript. We thank Tarek Baaboura, Daniel Burnier, and Philippe Rétornaz for help in building the hardware prototype. This work was supported by the Sixth Framework Programme of the European Community in the form of a Marie Curie Intra-European Fellowship (contract no. MEIF-CT-2006-040312). It reflects only the authors' views. The European Community is not liable for any use that may be made of the information.

References

1. Pfeifer, R., Bongard, J.C.: How the body shapes the way we think. MIT Press, Cambridge (2006)
2. Floreano, D., Mondada, F.: Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In: Proc. of the 3rd Int. Conf. on Simulation of Adaptive Behavior (SAB 1994), pp. 421–430. MIT Press, Cambridge (1994)
3. Thompson, A.: Artificial evolution in the physical world. In: Embley, D.W. (ed.) ER 1997. LNCS, vol. 1331, pp. 101–125. Springer, Heidelberg (1997)
4. Sims, K.: Evolving virtual creatures. In: Proc. of the 21st Annu. Conf. on Comput. Graphics and Interactive Tech. (SIGGRAPH), pp. 15–22. ACM Press, New York (1994)
5. Sims, K.: Evolving 3D morphology and behavior by competition. Artif. Life 1(4), 353–372 (1994)

6. Taylor, T., Massey, C.: Recent developments in the evolution of morphologies and controllers for physically simulated creatures. Artif. Life 7(1), 77–87 (2000)
7. Funes, P., Pollack, J.: Evolutionary body building: Adaptive physical designs for robots. Artif. Life 4(4), 337–357 (1998)
8. Lipson, H., Pollack, J.B.: Automatic design and manufacture of robotic lifeforms. Nature 406(6799), 974–978 (2000)
9. Chou, H., Reggia, J.A.: Emergence of self-replicating structures in a cellular automata space. Physica D 110(3-4), 252–276 (1997)
10. Dittrich, P., Banzhaf, W.: Self-evolution in a constructive binary string system. Artif. Life 4(2), 203–220 (1998)
11. Watson, R.A., Ficici, S.G., Pollack, J.B.: Embodied evolution: Distributing an evolutionary algorithm in a population of robots. Robot. Auton. Syst. 39(1), 1–18 (2002)
12. Bianco, R., Nolfi, S.: Toward open-ended evolutionary robotics: Evolving elementary robotic units able to self-assemble and self-reproduce. Connection Science 16(4), 227–248 (2004)
13. Channon, A.: Unbounded evolutionary dynamics in a system of agents that actively process and transform their environment. Genet. Program. Evolvable Mach. 7(3), 253–281 (2006)
14. Studer, G., Lipson, H.: Spontaneous emergence of self-replicating structures in molecube automata. In: Proc. of the 10th Int. Conf. on the Simulation and Synthesis of Living Systems (Artificial Life X), pp. 227–233. MIT Press, Cambridge (2006)
15. Smith, A., Turney, P., Ewaschuk, R.: Self-replicating machines in continuous space with virtual physics. Artif. Life 9(1), 21–40 (2003)
16. Whitesides, G.M., Boncheva, M.: Beyond molecules: Self-assembly of mesoscopic and macroscopic components. Proc. Natl. Acad. Sci. U.S.A. 99(8), 4769–4774 (2002)
17. Groß, R., Dorigo, M.: Self-assembly at the macroscopic scale. Proceedings of the IEEE 96(9), 1490–1508 (2008)
18. Penrose, L.S., Penrose, R.: A self-reproducing analogue. Nature 179(4571), 1183 (1957)
19. Jacobson, H.: On models of reproduction. Am. Sci. 46, 255–284 (1958)
20. Zykov, V., Mytilinaios, E., Adams, B., Lipson, H.: Self-reproducing machines. Nature 435(7039), 163–164 (2005)
21. Griffith, S., Goldwater, D., Jacobson, J.M.: Self-replication from random parts. Nature 437(7059), 636 (2005)
22. Christensen, A.L., O'Grady, R., Dorigo, M.: SWARMORPH-script: A language for arbitrary morphology generation in self-assembling robots. Swarm Intell. 2(2-4), 143–165 (2008)
23. Breivik, J.: Self-organization of template-replicating polymers and the spontaneous rise of genetic information. Entropy 3(4), 273–279 (2001)
24. Magnenat, S., Waibel, M., Beyeler, A.: Enki: The fast 2D robot simulator (2009), http://home.gna.org/enki/
25. Bishop, J., Burden, S., Klavins, E., Kreisberg, R., Malone, W., Napp, N., Nguyen, T.: Programmable parts: A demonstration of the grammatical approach to self-organization. In: Proc. of the 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3684–3691. IEEE Computer Society Press, Los Alamitos (2005)
26. White, P., Zykov, V., Bongard, J., Lipson, H.: Three dimensional stochastic reconfiguration of modular robots. In: Proc. of the 2005 Robotics: Science and Systems Conf., pp. 161–168. MIT Press, Cambridge (2005)

Acquisition of Adaptive Behavior for Virtual Modular Robot Using Evolutionary Computation

Keisuke Yoneda, Ikuo Suzuki, Masahito Yamamoto, and Masashi Furukawa

Hokkaido University, Laboratory of Autonomous System Engineering,
9 Nishi, 14 Kita, Kita-ku, Sapporo, 060-0814, Japan
{keisuke-y,ikuo,masahito,mack}@complex.eng.hokudai.ac.jp
http://autonomous.complex.eng.hokudai.ac.jp/

Abstract. In areas such as evolutionary robotics and artificial life, simulating artificial robots and organisms are significant challenges to acquire their proper behaviors that achieve given tasks. This study proposes Animated Robot ("Anibot"), which can behave by obeying physical laws in a virtual 3D environment. Especially, we aim to obtain a control system in evolution which makes it possible to behave "Anibot" autonomously. This paper focuses on a virtual modular robot with a flexible structure and simulating it for learning and controlling. The experimental results show that the modular robot can move toward a light source as its goal in different circumstances. In addition, we discuss an adaptive ability in the different circumstances and a motion mechanism of an obtained behavior.

Keywords: Physics-modeling, Learning, Neural network, Modular robot.

1 Introduction

In areas such as evolutionary robotics and artificial life, designing a body shape and controlling behavior for autonomous robots have been studied actively. Recently, the designing must take consideration in interaction between robots and their surrounding environment. There are many studies to acquire autonomous behaviors for them by computer simulations [1,2,3]. Sims [1] proposed a method for creating a virtual creature. In this methodology, the geometric morphology for a model structure and a neuro system for controlling a creature are both generated automatically using the genetic algorithm (GA). Therefore, it is expected that the creature adapts to the surrounding environment by use of physical simulation. Similarly, in the area of artificial life, behavior emergence and evolution of the artificial creature on a computer are significant problems.

This study has focused on a computer simulation that aims to have robots achieved given tasks autonomously for a virtual robot in a specific virtual 3D environment. For the purpose of this simulation, we have proposed Animated Robot ("Anibot") and developed a modeling tool to design "Anibot" [4]. "Anibot" is a virtual autonomous robot and it can behave by obeying physical laws in the virtual

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 181–188, 2011.

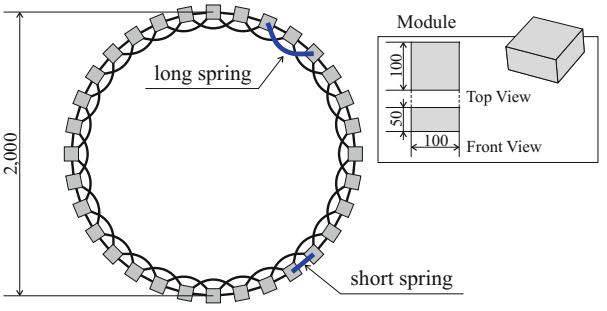


Fig. 1. Body system of modular robot (plan view)

3D environment. A difference between "Anibot" and a conventional animation is that "Anibot" has not only sensors, actuators to control autonomous behavior, but also physical properties such as a mass, a material property and so on.

Especially, this study focuses on a modular robot which is linked by simple modules, and its experiment to acquire a control system. A feature of the modular robot is that its behavior is obtained by an each module movement based on local communications between linked modules. In many studies, it is attracted how to control each modules autonomously [5,6,7,8]. Its self-reconfigurable property is also attracted. Murata et al.[6] develop a self-reconfigurable modular robot M-TRAN. M-TRAN is the modular robot which is connected by lattice type modules. This robot can change their module structure by ducking and separation. Therefore, it is expected for robots to adapt to several environments by changing its configuration. It has also a self-repairing ability in troubling some module by replacement of disabled parts with spare modules. Controlling the modular robot is generally studied by use of a rule-based control which is described by the specific behavior rule for each module. However the rule-based controlling approach has only allowed the robots relatively to control simple motion at present, for instance locomotion on a flat or an irregular ground, climbing over the wall and so on. This study adopts an evolutionary heuristic approach to control a modular robot. In particular, if this evolutionary approach is confirmed to generate distributed functions from an acquired behavior, it is expected that this approach is useful for a modular robot control [9].

The rest of this paper is composed as follows. Section 2 explains a virtual modular robot to examine its behavior. Section 3 illustrates learning experiments to acquire behaviors in moving it toward a goal, a light source, and shows some experimental results. Section 4 discusses their results and Section 5 concludes this study with some remarks and gives some directions toward the future work.

2 Modular Robot

2.1 Morphology

An intended modular robot is modeled by connecting rectangular modules with spring joints (Figure 1). It behaves by maintaining an initial state topology since

Table 1. Properties of shape and material for modular robot

	type	value
module	number of modules	32
	density [kg/m^3]	2,700
	restitution coeff.	0.300
	dynamic friction coeff.	0.400
	static friction coeff.	0.600
joint	number of short springs	32
	number of long springs	32
	natural length (short) [m]	0.200
	maximum length (short) [m]	0.400
	minimum length (short) [m]	0.050
	natural length (long) [m]	0.450
	maximum length (long) [m]	0.900
	minimum length (long) [m]	0.1125
	spring factor [N/m]	500
	damping factor [Ns/m]	0.400

Table 2. Input-output parameters of neuro controller

	variable
input	$\boldsymbol{v}_A$ (velocity vector of A)
	$\boldsymbol{v}_B$ (velocity vector of B)
	$\boldsymbol{u}_{AL}$ (unit orientation vector from $\boldsymbol{p}_A$ to $\boldsymbol{p}_L$)
	$\boldsymbol{u}_{BL}$ (unit orientation vector from $\boldsymbol{p}_B$ to $\boldsymbol{p}_L$)
	$\boldsymbol{u}_{AO}$ (orientation vector from $\boldsymbol{p}_A$ to $\boldsymbol{p}_{O_a}$)
	$\boldsymbol{u}_{BO}$ (orientation vector from $\boldsymbol{p}_B$ to $\boldsymbol{p}_{O_b}$)
	l (current length of spring i)
	l_0 (natural length of spring i)
	v_i (current elastic velocity of spring i)
output	A_i (amplitude for spring i)
	ω_i (angular frequency for spring i)

this study does not consider a topology change. Each module is connected to four modules surrounding it via four springs and is likely to behave itself by push and pull movements using the friction with ground. The behavior of this model is controlled by manipulating elastic velocities of each spring. Accordingly, all modules of this model move by propagating spring forces to the whole modules efficiently. In addition, Table 1 shows physical properties used for this model.

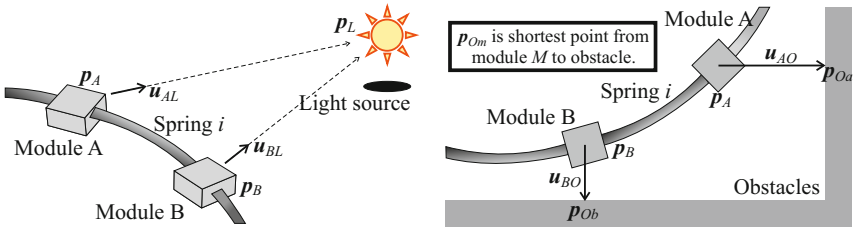
2.2 Control System

This model makes a movement using mainly an expansion and a contraction of springs. The elastic velocities of each spring are calculated by Eq. (1).

$$v_i(t + \Delta t) = v_i(t) + A_i(t)\sin(\omega_i(t)\Delta t + \theta_i(t)) \tag{1}$$

where $v_i(t)$ is an elastic velocity of spring i at time t, $A_i(t)$ is an amplitude, $\omega_i(t)$ is an angular velocity, $\theta_i(t)$ is an accumulated phase ($\theta_i(0) = \phi_i$, $\theta_i(t + \Delta t) = \theta_i(t) + \omega_i(t)\Delta t$) and ϕ_i is a specific initial phase of spring i. If $v_i(t) > 0$, the spring is expanded, if $v_i(t) < 0$, it is shrunk. Thus, the behavior of the whole model is controlled by manipulating $A_i(t)$ and $\omega_i(t)$ for each spring i. The movement of spring is controlled by neuro controllers equipped with springs. This study adopts an approach that each neuro controller actuates each spring independently.

This study deals with an easy task that the model mainly moves toward a light source as its goal. The modular robot is set to acquire proper behaviors in evolution for achieving given tasks. Therefore, their neuro controllers are defined as an Artificial Neural Network (ANN). These controllers are input by sensor signals perceiving a direction of the light source and obstacles and a situation of



(a) Light sensor of modules (b) Obstacle sensor of modules

Fig. 2. Sensor system of modular robot

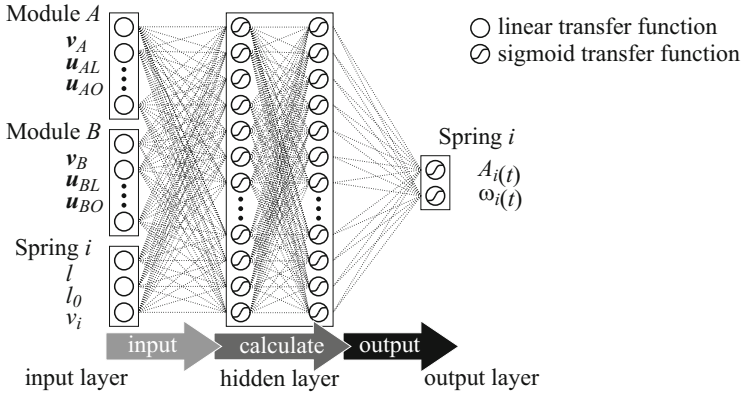


Fig. 3. ANN structure of neuro controller

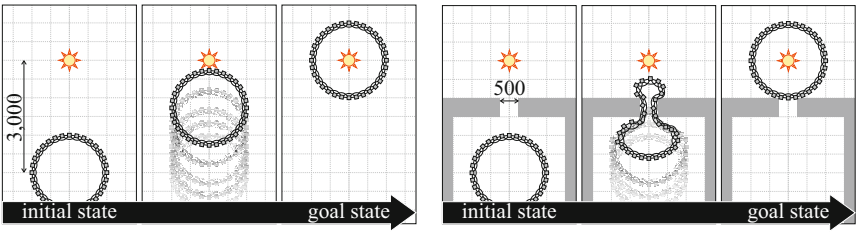
the actuator and it outputs some parameter values to calculate an elastic velocity of their spring (Table 2). The light sensor to perceive the goal is installed on each module. Additionally, each module has sensors which measure its velocity and perceive obstacles around it (Figure 2). Figure 3 shows a configuration diagram of an ANN which has a feed-forward network with hierarchic structure.

3 Simulation Experiments

Simulation experiments are carried out to achieve tasks for a given model described in previous section. Specifically, two tasks are described as follows:

1. move toward a light source on a flat ground (Figure 4(a)).
2. move toward a light source through a circumstance surrounded by several obstacles (Figure 4(b)).

These experiments examine an adaptive ability in different circumstances. Especially, in the 2nd experiment when the light is hidden by obstacles between a module and a light source, its module cannot perceive the goal. Thus the 2nd experiment is more difficult than the 1st one.



(a) A task of experiment 1st (b) A task of experiment 2nd

Fig. 4. A task of experiment (top view)

Table 3. Experimental condition

	type	value
GA	number of populations	20
	number of generations	200
	probability of crossover	0.900
	probability of mutation	0.010
	probability of inversion	0.300
simulation	step time [sec]	1/60
	the number of steps in simulation	8,200
	weighting coeff. α in Equation (3)	0.8

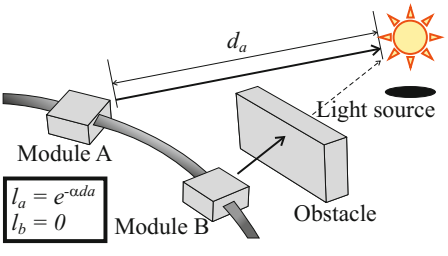


Fig. 5. Behavior evaluation of model

3.1 Experimental Condition

The purpose of experiments is to acquire proper behaviors that accomplishes given tasks. Actuators determine the behaviors. Each actuator has a controller implemented with ANN. We set that all ANN has the same synaptic weights. Differences among actuators are initial phases. Therefore, the synaptic weights assigned to one ANN and initial phases of all actuators are optimized to acquire the proper behaviors. A real number type of GA is adopted for this optimization. Experimental conditions for GA and simulation are shown in Table 3. All experiments described under-below are assumed to use these conditions.

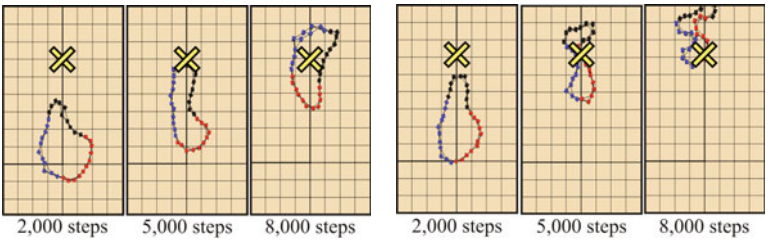
Obtained behaviors are evaluated by Eq. (2)

$$f = \sum_{t=0}^{N_s} \sum_{m=1}^{N_m} l_m \tag{2}$$

where

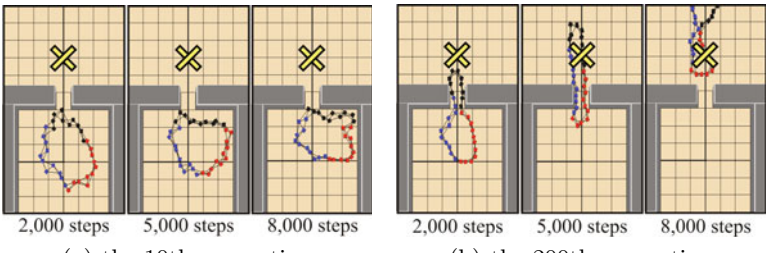
$$l_m = \begin{cases} e^{-\alpha d_m} & (if \quad \text{module } m \text{ receives light}) \\ 0 & (otherwise) \end{cases} \tag{3}$$

d_m is a distance measured from the m-th module to a light source, l_m is a light intensity which the m-th module receives from the light source, N_s is the number of steps a simulation, and N_m is the number of modules of which a virtual robots consists. In Eq. (3), α is a constant. Eq. (2) plays a role of a fitness function



(a) the 10th generation (b) the 200th generation

Fig. 6. Experiment 1st results



(a) the 10th generation (b) the 200th generation

Fig. 7. Experiment 2nd results

in GA and it evaluates how much all modules receive accumulated intensities during one simulation. A photo-tactic behavior is evaluated by Eq. (2). If the m-th module does not receive a light, a behavior of the m-th module is not evaluated (Figure 5). Finally results that GA optimizes so as to maximize Eq. (2) become obtained behaviors.

3.2 Experimental Results

Ten trials are attempted in each experiment under the same condition. Figure 6 and 7 show snapshots of behaviors for the best trials obtained in two experiments, respectively. The obtained behaviors of the model at the 10th and 200th generations in both experiments are shown. In each generation, motions at 2,000, 5,000 and 8,000 steps are drawn. In these snapshots, "X" mark indicates a position of a light source. It is confirmed that the model can arrive at a light source along the elapsed time. For these experiments, moving images are put on our website [10]. Moreover, Figure 8 and 9 show diagrams with the evaluated value of behaviors along the vertical axis and the number of generation along the horizontal axis in both experiments. The evaluated values for the best, worst and average trial are plotted in both diagrams. From both results, it is confirmed that a highly evaluated value is obtained as the GA generation is updated. In the 1st experiment, it is seen that the model can arrive at the goal at a low number generation (Figure 6(a)). In particular, in the behaviors obtained at the 200th generation, it is confirmed that three movements, such as start, move and stop, emerged. In the 2nd experiment, it is hardly seen that the model can pass

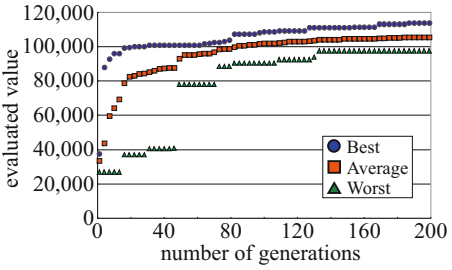


Fig. 8. Evaluated value in each generation for experiment 1st

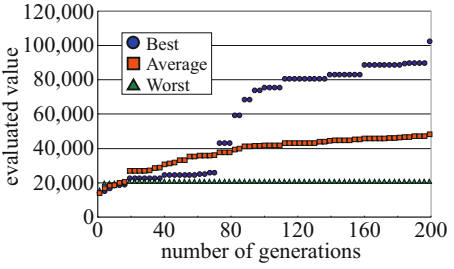


Fig. 9. Evaluated value in each generation for experiment 2nd

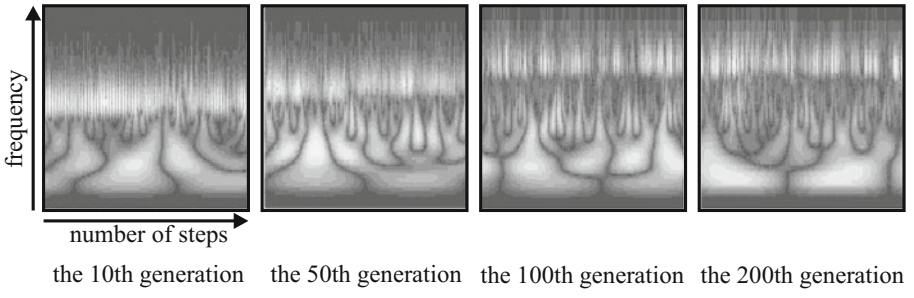


Fig. 10. Wavelet analysis results for spring frequency

through the gate by about the 100th generation because obstacles sometimes interrupt a light against the module. However, as the GA generation is updated, it is seen that behaviors are capable of achieving a given task.

4 Discussion

This study aims to generate motion rules in evolution for a virtual modular robot. In this section, we introduce time-frequency analysis into spring movements of the obtained behavior as a basic analysis. The analysis method is a wavelet transformation which makes it possible to observe spring length changes through time. Figure 10 shows analysis results for an arbitrary spring at the 10th, 50th, 100th and 200th generations. The horizontal axis and the vertical axis show an elapsed time and a frequency, respectively. The spring length frequency-resolved by the wavelet transformation appears in gray color. As the color changes from black to white gradually, the spring length increases larger. It is observed that there are remarkable differences at a high frequency range on the spring length among generations. As the generation number proceeds, a large spring length in this range appears at a higher frequency. This means the evolved neuro-controller diminishes unnecessary spring vibration. However, it is not identified why this phenomenon is caused. We need to investigate this phenomenon in more detail as a future work.

5 Conclusions and Future Work

We focus on a virtual modular robot and its physical simulation. Emergence of behaviors for the virtual modular robot in simulation can be regarded as a learning problem how the robot acquires the proper behavior. Evolutionary computation is a successful approach to this learning problem. The results prove that the defined modular robot can acquire behaviors in two different circumstances. The wavelet transformation give us some information on the whole motion. However, since we adopt a module unit with an actuator which works locally, it is difficult to analyze the whole model motion. It is too complex to analyze the obtained behavior. The followings are rest as some challenges in a future work.

1. Analyzing a motion mechanism that the model generates a locomotion.
2. Considering to a synchronous phenomenon in changing of each spring length through time.
3. Examination of a propagation mechanism generated as an elastic motion between neighboring springs.
4. Extracting control rules to move the robot from our experimental analysis.

References

1. Sims, K.: Evolving virtual creatures. In: SIGGRAPH 1994: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, pp. 15–22. ACM, New York (1994)
2. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes: autonomous locomotion, perception, behavior, and learning in a simulated physical world. Artif. Life 1(4), 327–351 (1994)
3. Boids background and update by craig reynolds, http://www.red3d.com/cwr/boids
4. Yoneda, K., Iwadate, K., Suzuki, I., Yamamoto, M., Furukawa, M.: Development of modeling tools for animated robot. In: Fourth International Symposium on Adaptive Motion of Animals and Machines (2008)
5. Murata, S., Kurokawa, H.: Self-reconfigurable robots. IEEE Robotics & Automation Magazine 14(1), 71–78 (2007)
6. Murata, S., Kakomura, K., Kurokawa, H.: Toward a scalable modular robotic system. IEEE Robotics & Automation Magazine 14(4), 56–63 (2007)
7. Rus, D., Vona, M.: Crystalline robots: Self-reconfiguration with compressible unit modules. Autonomous Robots 10(1), 107–124 (2001)
8. Kubica, J., Casal, A., Hogg, T.: Complex behaviors from local rules in modular self-reconfigurable robots. In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2001, vol. 1, pp. 360–367 (2001)
9. Kamimura, A., Kurokawa, H., Yoshida, E., Murata, S., Tomita, K., Kokaji, S.: Automatic locomotion design and experiments for a modular robotic system. IEEE/ASME Transactions on Mechatronics 10(3), 314–325 (2005)
10. Autonomous system engineering laboratory, hokkaido university, http://autonomous.complex.eng.hokudai.ac.jp/researches/physics-modeling/movies/yoneda/

Guiding for Associative Learning: How to Shape Artificial Dynamic Cognition

Kristen Manac'h and Pierre De Loor

UEB – ENIB-CERV – LISyC
Brest – France
{manach,deloor}@enib.fr

Abstract. This paper describes an evolutionary robotics experiment, which aims at showing the possibility of learning by guidance in a dynamic cognition perspective. Our model relies on Continuous Time Recurrent Neural Networks and Hebbian plasticity. The agents have the ability to be guided by stimuli and we study the influence of a guidance on their external behavior and internal dynamic when faced with other stimuli. The article develops the experiment and presents some results on the dynamic of the systems.

1 Positioning

Works in cognitive science show that cognition comes from the interaction between brain, body and environment [13,11,1]. Then, a cognitive system can be considered as an autonomous complex system disturbed by the environment, whose representation lies in sensory-motor invariants. This leads to different theoretical proposals in robotics and in artificial life [7,2,4]. Following these perspectives, some applications propose the use of dynamic systems [9,8,6]. Through interactions, these systems evolve between attractors and present plenty of different possible evolutions. These dynamic properties are the basis for adaptation, decision, memorization and also creation processes [12]. Different works address the sensory-motor invariants acquisition in such systems by pregiven adaptive behavior using Continuous Time Recurrent Neural Networks (CTRNN) which can approximate any dynamic system [5]. For example, [3] build photo-taxic robots which can adapt to sensors inversion at an ontogenetic scale.

We address the problem of learning behavior for such a system thanks to a specific interactive loop : a guidance signal. Indeed, learning by guidance is a more complex kind of adaptation because it can lead to different final behaviors. It is characterized by irreversibility which is a crucial property for ontogeny. Consequently, our concern is to establish experiments which will enable us to move from the status of self-adaptation to that of evolution induced by training, while preserving the use of an artificial dynamic cognition approach. However, evolutionary approaches are confronted with what [4] call the *hard problem of the enactive artificial intelligence* because it is necessary to associate phylogenetic mechanisms with the clarification of ontogenetic principles.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 189–196, 2011.

Before starting our explanation, we must mention that our work is similar to [6] who studies associative learning in evolved CTRNN. However, it is different because we use guidance to initiate the learning instead of discrete reward. Indeed, guidance is an evolutive mean to interact with the agent and to shape progressively its behavior. Here, we can use the metaphor of a child learning to ride a bike while being guided by an adult. If the child keeps his equilibrium, the adult stops to guide him progressively. Inversely, if the child loses his balance, the adult holds him. Progressively, the child will learn a new ability.

The section 2.1 presents our base experiment of learning by guidance. Models are then described in section 2.2 and section 2.3 presents the genetic algorithm used to set the parameters. The results are presented in the section 2.4.

2 Experiment

2.1 Principle

Let us consider an entity equipped with sensors functionally comparable to "eyes and ears". Its "ear" detects signals that make it change its orientation (right/left). Each "eye" detects the presence of one kind of light (A) or (B), but it initially does not display any particular behavior in presence of these lights. The signal sent to the ear will act as a guidance, so that an association between the signal received by the eyes and the one received by the ears is carried out dynamically in the interaction. In order to do so, we associate the presence of a light with a guidance moderated according to the effective behavior (for example: send signal turn to the right when light (A) is present and the entity is not turning to the right). The goal is that this guidance can attenuate gradually and that in the long term will no longer be necessary.

We study the behavioral enrichment at the individual level, i.e. at the ontogenetic level. Our artificial agents are equipped with a CTRNN. They have an effector to turn in both directions and three sensors that respectively detect a signal which corresponds to our guidance, the A light and the B light. The network is fully-connected. The agents must react to the guidance signal by turning to the right if the signal is negative and to the left if it is positive. We associate in an arbitrary way a light (A/B) with a side (left/right). The experiment consists in alternatively presenting the lights while guiding the agents according to the chosen association. The guidance signal is only present when the agent does not turn to the side associated with the light presented. However, a delay is introduced into the guidance mechanism to enable the dynamics of the system to take into account changes of perception. Thus, the guidance signal is not only delayed but also variable: it increases gradually if the agent behavior does not change. This adaptability leads us to speak about interactive guidance.

We seek to highlight the acquisition of a new behavior. This implies that the light presented does not condition *a priori* the behavior of the agents and thus that guidance must be necessary at the beginning of the experiment. If a new behavior conditioned by the light appears, then guidance became useless at the end of the experiment.

2.2 Model

The sensors mentioned previously and engines are coupled to the CTRNN. From then on, the term sensitive neuron will be used to designate a neuron which has an input coming from a sensor and the term motor neuron will be used to designate a neuron whose output is used by an effector. The cell potential y_i is governed by equation 1, where τ_i is the decay constant (range [0.5,2.1]), $w_{i,j}$ is the weight of synaptic connection from node i to node j (range [-8,8]), and I_i is an input from a sensor for a sensitive neuron. The firing rate z_i is calculated by using the equation 2, where b_i is the bias of the node i (range [-1,1]). The effector activation is computed by mapping the firing rate to the interval [-1,1] and multiplying by a gain (range [0,40]). In the same way, the input of the sensitive nodes is computed by multiplying the sensed value by a gain (range [0,40]). The sensed values are 1 or 0 on the "eyes" sensors depending on the presence of lights (A) and (B). For the "ear", the sensed value is negative for signal "turn to the left", positive for signal "turn to the right" and 0 for no signal.

$$\tau_i \frac{\delta y_i}{\delta t} = -y_i + \sum_j w_{j,i} z_j + I_i \tag{1}$$

$$z_i = \frac{1}{1 + e^{-(y_i + b_i)}} \tag{2}$$

$$\delta w_{i,j} = \eta(\alpha z_i z_j + \beta z_j + \gamma z_i) \tag{3}$$

The network plasticity, inspired from [14], corresponds to an hebbian rule (see equation 3), where η, α, β, γ are parameters (all in range [-1,1]) for each connection.

2.3 Parameters Setting

The parameters are: gains associated with the sensors and effector; for each neuron i, the decay constant τ_i and the bias b_i; for each connection, the parameters η, α, β and γ. They are determined by a genetic algorithm. The criteria to optimize by the algorithm is not a specific task but an ontogenetic development. The fitness used by the genetic algorithm is computed in 3 independent phases giving 3 scores : f_1, f_2 and f_3. In short, the three phases are:

1. Guidance reaction checking. During a trial, the entity is guided towards a side. The score of an entity for a trial is the time it has turned to the signaled side. This phase is made up of 20 trials alternating guidance towards the right and towards the left. Score f_1 is computed using equation 4.

$$f_1 = \min_{i \in [0.20]} \left\{ val_i | val_i = \frac{\int_{ts_i}^{te_i} \Delta_{Turn}(t)dt}{te_i - ts_i} \left(1 - \frac{\int_{ts_i}^{te_i} \Delta_{Energy}(t)dt}{te_i - ts_i} \right) \right\} \tag{4}$$

where val_i is the score of the trial i which starts at ts_i and stops at te_i. Δ_{Turn} is the fitness component corresponding to the correlation between the guidance signal and the motor activity given by the equation 5.

$$\Delta_{Turn}(t) = \begin{cases} 1 - \parallel M(t) - S(t) \parallel & : \quad M(t)S(t) > 0 \\ 0 & : \quad else \end{cases} \tag{5}$$

where $M(t)$ is the activity of the motor neuron mapped to the range [-1,1] and $S(t)$ is the guidance signal. Δ_{Energy} is used to avoid oscillating behaviors.

$$\Delta_{Energy}(t) = \begin{cases} 0 & : \quad M(t-1)M(t) > 0 \\ 1 & : \quad else \end{cases} \tag{6}$$

2. Guidance according to an association. Lights (A/B) are arbitrarily associated to sides (left/right). The duration of each light presentation is arbitrarily chosen in the range [1.5,4.5]. During a presentation, only one light is in the environment of the entity. After a time during which the behavior of the entity can stabilize (10 lights presentations), the guidance starts according to the association previously decided. This second phase is made up of 50 light presentations. For each presentation, the type of light is randomly chosen. Score f_2 is computed using equation 7.

$$\begin{aligned} f_2 = \quad & (\min\{Score_A, Score_B\}) \\ & \times \min_{i \in [10.50]} \left\{ val_i | val_i = \frac{\int_{ts_i}^{te_i} \Delta_{Turn2}(t)dt}{te_i - ts_i} \left(1 - \frac{\int_{ts_i}^{te_i} \Delta_{Energy}(t)dt}{te_i - ts_i} \right) \right\} \end{aligned} \tag{7}$$

Δ_{Turn2} is given by equation 8.

$$\Delta_{Turn2}(t) = \begin{cases} 1 & : \quad M(t)C(t) > 0 \\ 0 & : \quad else \end{cases} \tag{8}$$

where $C(t)$ stands for the association ($C(t) = 1$ when the presented light at time t is associated with side left and $C(t) = -1$ else). $Score_A$ and $Score_B$ are defined with equation 9.

$$Score_X = \frac{Es_X - Ee_X}{Es_X}, X \in \{A, B\} \tag{9}$$

$Score_X$ measures the progress of the agent in associating the previously chosen side to light X. Es_X and Ee_X are respectively the mean value of the score at the beginning of the experiment and the mean value at the end.

$$Es_X = \frac{\sum_{i=10}^{20} \frac{1}{te_i - ts_i} \int_{ts_i}^{te_i} \Delta_{Light}{}^X(t)dt}{\sum_{i=10}^{20} X(t)}, X \in \{A, B\} \tag{10}$$

$$Ee_X = \frac{\sum_{i=40}^{50} \frac{1}{te_i - ts_i} \int_{ts_i}^{te_i} \Delta_{Light}{}^X(t)dt}{\sum_{i=40}^{50} X(t)}, X \in \{A, B\} \tag{11}$$

where $X(t)$ is 1 when light X is present, 0 else, and ${\Delta_{Light}}^{X}(t)$ is

$$ {\Delta_{Light}}^{X}(t) = \begin{cases} S(t) & : \quad C(t)M(t) > 0 \quad and \quad X(t) = 1, X \in \{A, B\} \\ 0 & : \quad else \end{cases} \tag{12} $$

3. Guidance according to the inverse association. The association used in phase 2 is inverted, then the same process is used to get a score f_3. By carrying out this inversion, we want to make sure the score of the agent depends on guidance associated with the light and not on a predisposition of the agent.

Between each phase, the neural network is reset. The final score of an agent is given by equation 13. A "good" agent is an agent which one can immediately guide and which is able to take into account a guidance to progressively associate a light with a side. The algorithm preserves the best individuals, in the proportion of a third. The second third is obtained by cloning the best individuals, each one of these clones systematically undergoing a mutation. All the parameters have the same mutation rate. The mutation consists in varying very slightly one of the parameters selected in a random way. The population is supplemented with a third of new individuals.

$$ f = a * f_1 + b * min(f_2, f_3) \tag{13} $$

2.4 Results

The results presented here were obtained by applying the genetic algorithm to populations of 50 individuals, controlled by 6 neurons networks, during 30,000 generations. Each agent has 208 parameters which are directly encoded into genes (real values). Figure 1 illustrates how the best agent found by the genetic algorithm reacts to a guidance scenario when no light is in the environment. One guidance pulse is always sufficient to change the motors direction, however it may be necessary to repeat this guidance to stabilize the motor on the expected side. Figure 2 illustrates the protocol we used to observe the performance of the

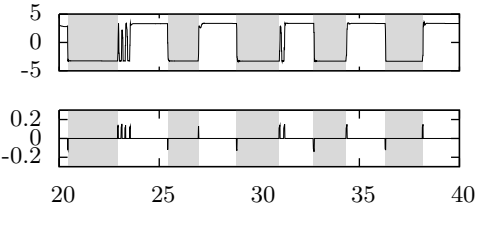


Fig. 1. Motor activity according to a guidance scenario. The first graph plots the motor activity and the second graph plots the guidance signal. Grey areas correspond to guidance towards the right whereas white areas correspond to guidance towards the left. The periods lasts are randomly chosen.

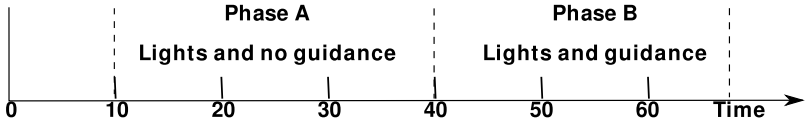


Fig. 2. Protocol used to observe the agents

evolved agents. The experimenter decides on an association light-side. Phase A starts at time 10 in order to allow the network to reach a stationary mode. Lights are presented alternatively and there is no guidance. It allows us to check the non correlation between the motor activity and the lights before the training. Phase B, starting at time 40, corresponds to the training according to the chosen association. Lights are presented to the agent. The experimenter guides it if it is necessary. If the agent progresses, the experimenter intervenes less and less.

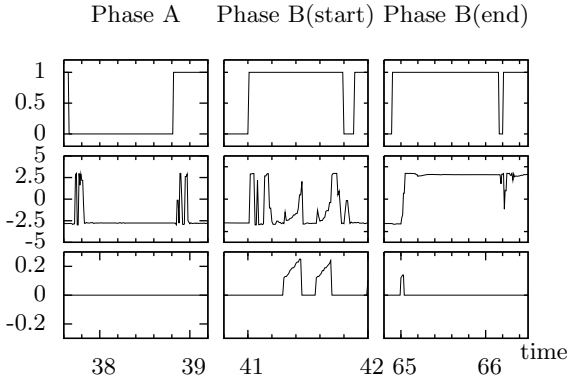


Fig. 3. Different time periods of one simulation show the ontogenetic evolution of the motor activity according to lights presentation

Figure 3 shows the activity of the best evolved agent interacting with a virtual experimenter using the protocol described by figure 2. The association chosen is light A/left - light B/right. From top to bottom, the first graph corresponds to light presentations: 0 for light A, 1 for light B. A positive activity on the motor neuron (second graph) means that the agent is turning to the left, a negative activity that it is turning to the right. The third graph plots the guidance. At the end of phase A (left part on Figure 3), changes of lights induce a change in the motor activity. However, there is no direct correlation between the motor and the lights even if the agent exhibits a tendency to turn to the right. At the beginning of phase B (central part of Figure 3), the guidance causes the motor to turn towards the desired side, but it is necessary to repeat the guidance. At the end of phase B (right part of Figure 3), there is no need to guide the agent anymore as the motor activity is correlated to the lights (Obviously, changing side of the motor takes a little time, during which some little oscillations occur).

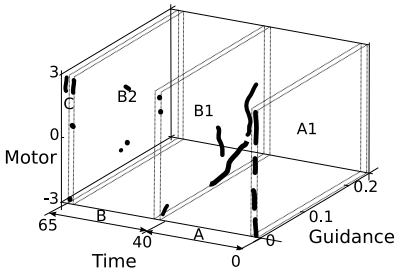


Fig. 4. Motor activity compared to guidance during some short time periods

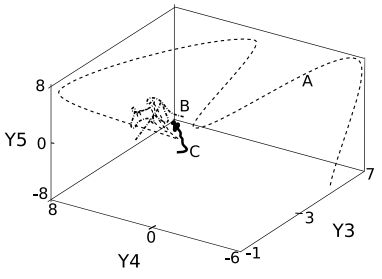


Fig. 5. Activity of the internal neurons (Y_4 and Y_5) compared to activity of the guidance sensitive neuron (Y_3)

Figure 4 plots the motor activity compared to guidance during short time periods taken on the whole episode (not all time periods for the sake of clarity), but only when light A is in the environment. Light A has to be associated to side left, that means a positive motor activity. Set A1 corresponds to a short interval extracted from phase A, *i.e.* before the training. During this period, the motor activity is distributed between negative and positive values. Set B1 corresponds to the beginning of the training phase. At this time, motor activity is more negative with a high need of guidance. Set B2 is extracted from the end of the training. Motor activity is concentrated mostly on positive values. Set C corresponds to the first presentation of light A after the training period. During the considered time, motor activity is positive. Figure 5 illustrates how neural dynamics is shaped by training. Dynamics of neural outputs y_3, y_4, y_5 are represented at stages of presentation of light A. Y_3 is the output of neuron $n3$ which owns the guidance signal as input. It is surprising to observe that during phase A, this output varies widely. During phase B (guidance), trajectory joins a smaller space. Finally, this space is approximately kept during phase C (end of learning). This dynamic can be interpreted as the fact that during phase A, the system is very *receptive* to stimuli coming from guidance. Oscillations allow to explore a lot of states of the system. At the opposite, during guidance, the system falls into another lesser extended attractor, corresponding to an association side/light. However, this attraction depends on the guidance signal which depends itself on an arbitrary choice of the association used during the experiment. Similar curves are obtained for the presentation of light B.

3 Prospects

This work has focused on the evolution of the dynamics of a CTRNN during an associative learning task involving guidance. We have shown how an external influence, *i.e.* the guidance, may impact the internal dynamics. From a dynamical approach of artificial intelligence, it addresses the problem of sensory-motor acquisition of a dynamic system at an ontogenetic scale. To follow an enactive-like

perspective, a long term goal is to reach co-evolutive mechanisms. This work is a contribution in this direction, as it addresses the question of influencing the drift of a dynamic system through a external signal. Obviously, the task presented here is voluntarily very simple because we are more interested in the training than in the learned behavior itself. To treat more complex tasks, it will be necessary to increase the sensory-motor capacities of the entity. For example, the design of an entity should require consideration of the entity's shape [10], the evolvability of the morphology of the neural network and the imagination of slight variations of the complexity of the task.

Acknowledgements. This work is supported financially by the Conseil Régional de Bretagne.

References

1. Beer, R.D.: Dynamical approaches to cognitive science. Trends in Cognitive Sciences 4(3), 91–99 (2000)
2. Brooks, R.A.: Intelligence without representation. Artificial Intelligence 47, 139–159 (1991)
3. Di Paolo, E.A.: Homeostatic adaptation to inversion in the visual field and other sensorimotor disruptions. In: Meyer, J.A., Berthoz, A., Floreano, D., Roitblat, H.L., Wilson, S.W. (eds.) From Animals to Animats 6. Proceedings of the VI International Conference on Simulation of Adaptove Behavior, pp. 440–449 (2000)
4. Froese, T., Ziemke, T.: Enactive artificial intelligence: Investigating the systemic organization of life and mind. Artif. Intell. 173(3-4), 466–500 (2009)
5. Funahashi, K., Nakamura, Y.: Approximation of dynamical systems by continuous time recurrent neural networks. Neural Networks 6, 801–806 (1993)
6. Izquierdo, E., Harvey, I., Beer, R.D.: Associative learning on a continuum in evolved dynamical neural networks. Adaptive Behavior 16(6), 361–384 (2008)
7. McMullin, B.: Thirty years of computational autopoiesis: A review. Artificial Life 10, 277–295 (2004)
8. Montebelli, A., Herrera, C., Ziemke, T.: On cognition as dynamical coupling: An analysis of behavioral attractor dynamics. Adaptive Behavior 16, 182–195 (2008)
9. Negrello, M., Pasemann, F.: Attractor landscapes and active tracking: The neurodynamics of embodied action. Adaptive Behavior 16, 196–216 (2008)
10. Pfeifer, R., Gomez, G.: Interacting with the real world: design principles for intelligent systems. Artificial Life and Robotics 9(1), 1–6 (2005)
11. Pfeifer, R., Bongard, J.C.: How the Body Shapes the Way We Think, A New View of Intelligence. MIT Press, Cambridge (2006)
12. Simao, J.: Aperiodic (Chaotic) Behavior in RNN with Homeostasis as a Source of Behavior Novelty: Theory and Applications. In: Recurrent Neural Networks, pp. 400–420 (2008)
13. Varela, F.J., Thompson, E., Rosch, E.: The Embodied Mind. MIT Press, Cambridge (1993)
14. Wood, R., Di Paolo, E.A.: New models for old questions: Evolutionary robotics and the 'A not B' error. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 1141–1150. Springer, Heidelberg (2007)

ALife in the Galapagos: Migration Effects on Neuro-Controller Design

Christos Ampatzis, Dario Izzo, Marek Ruciński, and Francesco Biscani

Advanced Concepts Team,
Keplerlaan 1 - 2201 AZ Noordwijk - The Netherlands
{Christos.Ampatzis,Dario.Izzo,Marek.Rucinski,Francesco.Biscani}@esa.int
http://www.esa.int/act

Abstract. The parallelization of evolutionary computation tasks using a coarse-grained approach can be efficiently achieved using the island migration model. Strongly influenced by the theory of punctuated equilibria, such a scheme guarantees an efficient exchange of genetic material between niches, not only accelerating but also improving the evolutionary process. We study the island model computational paradigm in relation to the evolutionary robotics methodology. We let populations of robots evolve in different islands of an archipelago and exchange individuals along allowed migration paths. We show, for the test-case selected, how the exchange of genetic material coming from different islands improves the overall design efficiency and speed, effectively taking advantage of a parallel computing environment to improve the methodology of evolutionary robotics, often criticized for its computational cost.

Keywords: Stochastic Optimisation, Evolutionary Robotics, Island Model, Parallel Computing.

1 Introduction

The theory of punctuated equilibria is a theory in evolutionary biology which essentially states that evolution proceeds with rapid changes after long periods of stasis [9]. The theory inspired Cohoon et al. [4] in formulating the island migration model, a coarse-grained parallelisation approach to global optimisation (GO). More specifically, the authors originally focused on the parallelization of genetic algorithms. Thus, the underlying mechanisms of this stochastic global optimization technique inspired by Darwinian evolution, namely recombination, mutation and selection, are complemented by migration. Initially isolated populations, while evolving in parallel, exchange individuals at a certain rate, thus interacting with each other. As a result, not only do GA populations evolve faster, but their performance is also improved. The island model paradigm has also been applied to the parallelisation of other evolutionary algorithms, as differential evolution (DE [11]) and applied on difficult and high dimensional global optimisation problems [5].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 197–204, 2011.

Evolutionary Robotics. In the past years the application of artificial evolution to the optimisation of neuro-controllers for autonomous robots has been gaining a lot of momentum. The approach called Evolutionary Robotics (ER) is, essentially, a methodological tool to automate the design of robots' controllers [10]. It is typically based on the use of artificial evolution to find sets of parameters for artificial neural networks that guide the robots to the accomplishment of their task. With respect to other design methods, ER has the theoretical advantage that it does not require the designer to make strong assumptions concerning what behavioural and communication mechanisms are needed by the robots.

However, to-date, the complexity of the tasks solved by agents controlled by evolved neuro-controllers is lower than the complexity achieved by other methods using hand-coded controllers driven by expert knowledge. Also, even if automatic techniques could in principle reduce the human effort required to design controllers, this is usually not the case [8]. In other words, the complexity achieved by automatic approaches seems incommensurate with the effort expended in setting up and configuring the evolutionary system. Therefore, despite the theoretical advantages of automating the design problem for autonomous agents, the robotics control community cannot yet claim to having reaped its benefits. More effort should be put by researchers in reducing the computation time required until a solution to a problem at hand is obtained with these techniques, and at the same time in creating a framework that can generate more complex solutions without a significant effort overhead on the side of the experimenter. Various approaches in the literature have explored the possibility of enhancing the efficiency of automatic design tools, such as incremental evolution and symbiotic evolution, but the focus of such methods is not on creating a generic and simple design framework and certainly not on the algorithmic side.

Evolution in robotic islands. Typically, ER researchers launch a number of independent evolutionary runs, each differentiated by a different initial random seed. Subsequently, for each run, the best individual of the last generation, or alternatively the best individual encountered throughout evolution is identified and analysed (post-evaluated). Thus, some of the runs end up successful, and others not. By successful run we mean a run where the fitness obtained is the maximum, or one where a neuro-controller is produced that is able to drive the robots to the accomplishment of their task. Every evolutionary run is characterised by the prevalence of a certain genetic material that gets spread through recombination and survives through selection. Variation of genetic material within the population comes through random mutation only.

This paper introduces the idea of adding the island model paradigm to the ER arsenal, thus using the island model to evolve artificial neural networks controlling robots. In this case, the experimenter will still be launching multi-start evolutionary runs, only this time, the dynamics of the stochastic search in a single run will not be fully determined by the initial random seed, but also co-determined by the migration among runs and information exchange. Populations will be invaded with genetic material shaped elsewhere. Migrants might represent solutions radically different than the solutions developed in a given population.

The analogies between the biological observations and the effect of migration in GO still hold when the application is the ER methodology. In particular, exploration arises from migration and exploitation from isolated evolution in islands. Finally, while in GO the island model introduces new local minima and optimisers to a pool of solutions, in ER migration represents introducing new ways of solving the task in a pool of existing solutions. This is because a solution is, apart from a fitness number, the behavioural capabilities of autonomous, simulated or real agents that are evaluated in a given environment and with respect to a given task. This paves the way to addressing more complex ER scenaria, including evolving populations for different tasks in different islands, and then letting migration perform the mixing of genetic material and behavioural capabilities.

2 Simulating Evolution in an Archipelago

The simulation environment we used for this study has been developed entirely by the European Space Agency's Advanced Concepts Team and is freely available as an open source project named Parallel Global Multiobjective Optimiser (PaGMO). The code, entirely written in C++ and tested on Windows XP, Ubuntu, and Leopard OS, implements the asynchronous island model paradigm for generic optimisation purposes. It defines Individuals, Populations, Islands and Archipelagi as C++ objects, providing an easy-to-use computational environment where to simulate the concurrent evolution of populations. PaGMO objects can also be instantiated directly from Python and each island can be assigned the same or a different algorithm to evolve its population. The evolution in each island is assigned to a different thread of the underlying operating system so that parallelisation is obtained when multiple processors are available. Communication (migration) between threads (islands) occur in an asynchronous fashion. An *archipelago* is defined by the islands it contains, but also by their geographical location which determines the possible migration routes an individual can take. We describe such migration routes as a graph where the nodes represent different islands and the edges represent the possible connections between islands. We refer to this graph as the migration topology. An *island* is defined by a population of individuals, by an evolutionary strategy (or a global optimisation algorithm) and by a task (or a global optimisation problem). A *population* is defined by the individuals it contains and, finally, an *individual* is defined by its chromosome. Such a generic framework facilitates studies on the island model impact on optimisation algorithms in general and population based meta-heuristics in particular.

Evolutionary robotics and stochastic global optimisation. Here we briefly describe ER problems and their mathematical structure, as encountered in the literature, trying to highlight those elements that make of ER problems a very special case of optimisation problems. We indicate with $\mathcal{N}(\mathbf{x})$ a generic robot controller tasked to translate robot sensory inputs into actuation commands. The controller is defined by a number of parameters we indicate

with $\mathbf{x} \in R^N$. The ultimate aim is choosing the best $\mathbf{x}$. To this aim, a function $f(\mathbf{x}, \mathbf{s})$ is constructed that maps the chromosome to a fitness value identifying how well the robot is able to solve a given task using the controller $\mathcal{N}(\mathbf{x})$ during a particular experiment defined by $\mathbf{s}$. Typically, $\mathbf{s}$ includes the robots' initial conditions/configuration, but also variables defining uncertain environment characteristics and sensory noise; $\mathbf{s}$ is a stochastic variable with known distribution. The objective function assigns a value to the decisions taken by the controller $\mathcal{N}$ both in terms of achieving or not the task required and in terms of their optimality. The generic ER problem can thus be written as:

$$\begin{array}{c} \text{find: } \mathbf{x} \in R^n \\ \text{to maximize: } J(\mathbf{x}) = E[f(\mathbf{x}, \mathbf{s})] \end{array} \tag{1}$$

where $E[\cdot]$ indicates the expected value of its argument. One wants to find a controller $\mathcal{N}$ allowing the robot to solve an assigned task for all (or almost all) possible realisations of the experimental set-ups $\mathbf{s}$ and that is optimal with respect to the expected value of the defined objective function. The ER problem described above shares a lot of similarities with problems studied in stochastic programming and indeed some of the techniques used there are also used by ER researchers. The sample average approximation (SAA) method [6], for example, may be used to approximate the objective function so that $J(\mathbf{x}) \approx \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{x}, \mathbf{s}_j)$ where a sample $\mathcal{S} = \mathbf{s}_1 ... \mathbf{s}_n$ is considered, where each $\mathbf{s}_j$ is an independent identically distributed random variable (expected value is the average) and has the same probability distribution as $\mathbf{s}$. The SAA is not an algorithm, but just an approximation method and one has still to solve the resulting problem (no longer stochastic) with an appropriate numerical procedure. Typically in ER an evolutionary algorithm is used to solve the SAA approximated problem (genetic algorithm or evolutionary strategies). This is not a necessary choice and global optimisation heuristics based on paradigms other than the darwinian evolution may, in principle, be able to solve the problem. Once the approximated problem is solved, its solution $\hat{\mathbf{x}}$ is a candidate solution to the original stochastic programming problem and its quality as such needs to be assessed. This is done by performing one evaluation of $J(\mathbf{x})$ using a different and typically larger sample $\mathcal{S}'$ (called in ER works post-evaluation of a solution).

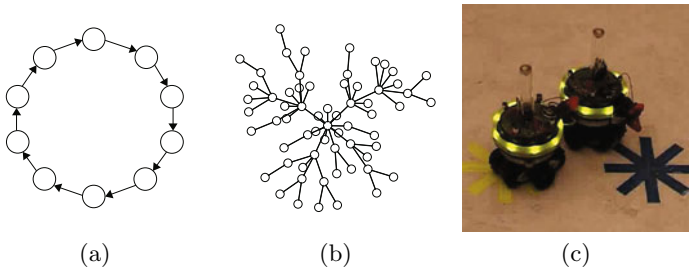
The evolutionary strategy adopted. In this paper we use differential evolution [11] as a global optimisation heuristic to solve the ER task considered. In PaGMO terms this means that each island will be assigned a population which will evolve, before each migration happens, via an instance of DE for N generations. The objective function for the evolution is the SAA approximation of the chosen task. The sample $\mathbf{s}$ (and thus the objective function) is changed every $m = N$ generations to avoid the optimiser to take advantage of particularities of one particular sample. One iteration of DE is made of three phases: mutation, crossover and selection. In the mutation phase, and for every individual in the population of size NP, a so-called mutant individual is formed. To create such a mutant we use the DE2 variant of the algorithm (see [11]):

$$v_{i,G+1} = x_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G});$$

where G is the generation number, r_{1-3} are 3 different, randomly selected individuals, and $F > 0$ is the constant amplification factor. Exponential crossover is then performed between the mutant and the original individual with a step probability CR. In the experiments, we used $F = 0.8$, $CR = 0.8$, $NP = 30$ and sample size $n = 40$, with the sample changing every $m = 50$ generations .

Migration and topology. Setting up an experiment which involves optimisation with the island model requires making cross-related choices about the following parameters [3]: (1) the *number of islands*, (2) the *archipelago topology* which defines feasible migration paths between islands, (3) the *migration rate* which tells how many individuals migrate at a time, (4) the *migration frequency* which determines how often migration occurs, and (5) the *migration algorithm*. The latter includes defining if the migration is synchronous (all islands are forced to migrate at the same time during which no optimisation is performed) or asynchronous (every island performs migration as soon as it is ready to do so), which individuals from the population migrate, how they are distributed among neighbour islands, and which (if any) individuals in the destination population are replaced by migrants and under what conditions. All these choices have an impact on the overall robustness of the archipelago, however fine-tuning these parameters is out of scope of this paper. That is why we decided to make choices commonly found in the existing literature (see for instance [7,12]), as this should be sufficient to demonstrate that the island model can enhance the potential of ER as an optimisation technique.

In our experiments, the archipelago consists of 10 islands connected with a one-way ring topology (see figure 1a). Of course, there is an infinite number of other possible choices which could include lattices, hypercubes, and even more complex networks, like small-world Barabasi-Albert model networks (see figure 1b) for large numbers of islands. We decided to use an asynchronous migration algorithm as it is more suitable to model modern large-scale distributed computational environments in which synchronisation of all nodes is either impossible or extremely expensive. Every island migrates its best individual every 50 generations of DE with probability 0.4. On arrival, the migrating individual replaces a random individual in the destination population if it has a better fitness value.



(a) (b) (c)

Fig. 1. (a) Example of a ring topology with 10 nodes; (b) a Barabasi-Albert model topology with 64 nodes; (c) a snapshot of the robot and the task

The following pseudo-algorithm defines the computations performed by each island (thread):

1: create a starting population $\mathcal{P}_i$ of dimension NP
2: while !*stopping-criteria*
3: generate a sample $\mathbf{s}$ and the SAA approximated problem
3: evolve $\mathcal{P}_i$ for N generations using DE
5: migrate the best individual to the neighbour island
6: insert the migrating individual from the neighbour island into $\mathcal{P}_i$ (if available)

3 Test Bed: Evolving Self-assembling Robots

The task we consider as a test-bed is described in detail in [1,13] and it can be roughly summarised as follows. Two robots are initialised at a given distance between a lower and an upper bound with given initial orientations; the robots do not have means for explicit communication; they are only able to sense their distance to their group mate. The agents should coordinate their movements in order to allocate roles and connect to each other via the gripping mechanism (see figure 1c). The controllers are evolved in a simulation environment which models some of the hardware characteristics of the *s-bot*, a small mobile autonomous robot with self-assembling capabilities. The network $\mathcal{N}(\mathbf{x})$ used to control the robots is a Continuous Time Recurrent Neural Network (CTRNN [2]), with a fixed feed-forward architecture comprising an arrangement of 11 input neurons, 10 hidden neurons and 3 output neurons. Decay constants (τ), connection weights between two neurons (ω) and bias terms (β) are genetically specified parameters. Each CTRNN is thus encoded by a vector $\mathbf{x}$ comprising 263 real values. A random population of CTRNNs is generated by initialising each component of $\mathbf{x}$ to values randomly chosen from a uniform distribution in $[-10, 10]$.

The control is homogeneous since the robots share the same dynamical controller (i.e., the CTRNN). We consider the problem in the form of Eq.(1) where the function $\mathbf{f}(\mathbf{x}, \mathbf{s})$ assesses the ability of the two robots to approach each other and physically assemble through the gripper, without interfering or dictating the role allocation strategy the robots should use. The experimental conditions $\mathbf{s}$ include the robots' initial conditions (i.e. orientations and mutual distance) and the sensory noise. The value of $\mathbf{f}(\mathbf{x}, \mathbf{s})$ can vary between 0 and 100. The maximum value corresponds to collision-free successfull robot self-assembly. The samples $\mathcal{S}$ used to obtain the SAA approximation to the problem contain 40 experimental conditions $\mathbf{s}_j$ generated consistently with [1,13].

4 Experiment and Results

The problem defined above has been solved in [1,13], where an evolutionary strategy ((μ, λ)-ES) was also employed, and the resulting neuro-controller was

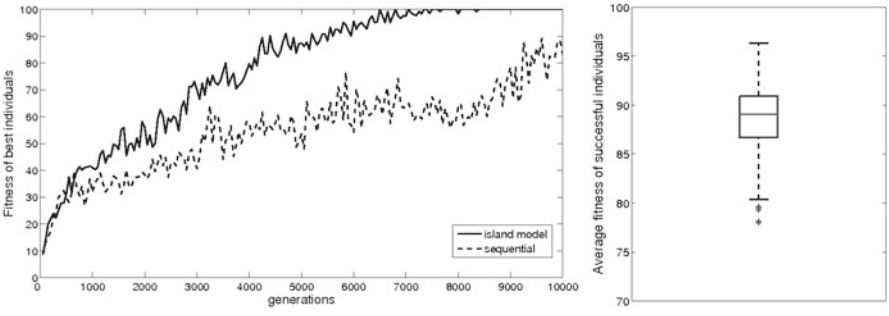


Fig. 2. (a) Best fitness across generations for sequential and island model setups; (b) Boxplot of the average fitness of all successful individuals after post-evaluation

successfully transfered to real robots. In order to study the effects of the island model and migration in particular, we create an archipelago of ten islands containing different populations of 30 CTRNN. We allow each population to evolve with DE in each island for 10,000 generations, i) with no migration (this is equivalent to a set of 10 independent sequential runs of DE), and ii) allowing migration using a ring topology as described in section 2.

The comparison is done with respect to the maximum fitness achieved during evolution and in terms of the number of generations elapsed up to that point. The maximim fitness is defined as the best fitness across all individuals and across all the islands. In figure 2a we can see the results for the two experiments. When no migration is allowed (sequential algorithm), DE manages to reach around 80% of the maximum fitness. When migration takes place, the island model implementation of DE does reach the value of 100.[1]

The quality of the candidate solutions found $\hat{\mathbf{x}}$ needs to be assessed once the evolution has completed as explained previously. This is needed to exclude the possibility that the solutions discovered have taken advantage of favourable conditions during evolution. We post-evaluate successful individuals (with fitness 100) on a set $\mathcal{S}'$ of 1,000 experimental conditions, all differing with respect to the random noise applied on sensors/actuators, initial robot positions and orientations. In figure 2b we show a boxplot of the average fitness obtained by all successful individuals in post-evaluations. The best scoring individual in these tests achieved an average fitness score over 96, which confirms that the quality of the solution obtained with the presented framework is very high across almost all possible realisations of the experimental set-up $\mathbf{s}$. Still, as we can see from the boxplot, this is not the case with all individuals that scored maximum during evolution, as some achieve considerably lower fitness values in post-evaluations.

[1] Notice that for the sequential case we have prolonged the evolution for 5,000 generations more but without achieving the maximum—results not shown on the graph.

5 Conclusion

In this article, we have introduced the island model paradigm to the evolutionary robotics methodology. We have presented one test case where migration of genetic material across evolving populations not only accelerates evolution but also creates better individuals, managing to obtain solutions to the problem at hand. Future work will include a more thorough understanding of how the parallel version improves the algorithm performance, the effect of migration on the population diversity, the generation and analysis of a more statistically sound sample of experiments, as well as the exploitation of the island model for solving more complex tasks with ER.

References

1. Ampatzis, C., Tuci, E., Trianni, V., Christensen, A.L., Dorigo, M.: Evolving self-assembly in autonomous homogeneous robots: experiments with two physical robots. Artificial Life 15(4), 1–20 (2009)
2. Beer, R.D., Gallagher, J.C.: Evolving dynamical neural networks for adaptive behavior. Adaptive Behavior 1, 91–122 (1992)
3. Cantu-Paz, E.: Efficient and Accurate Parallel Genetic Algorithms. Kluwer Academic Publishers, Norwell (2000)
4. Cohoon, J., Hegde, S., Martin, W., Richards, D.: Punctuated equilibria: a parallel genetic algorithm. In: Proceedings of the Second International Conference on Genetic Algorithms and Their Application, pp. 148–154. L. Erlbaum Associates Inc., Hillsdale (1987)
5. Izzo, D., Ruciński, M., Ampatzis, C.: Parallel global optimisation meta-heuristics using an asynchronous island-model. In: Proceedings of CEC 2009 (2009)
6. Kleywegt, A.J., Shapiro, A., Homem de Mello, T.: The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization 12(2), 479–502 (2002)
7. Martin, W.N., Lienig, J., Cohoon, J.P.: Island (migration) models: evolutionary algorithms based on punctuated equilibria, ch. C6.3:1-C6.3:16. Institute of Physics Publishing, Bristol (1997)
8. Mataric, M., Cliff, D.: Challenges in evolving controllers for physical robots. Robotics and Autonomous Systems 19(1), 67–83 (1996)
9. Eldredge, N., Gould, S.J.: Punctuated Equilibria: An Alternative to Phyletic Gradualism. Freeman, Cooper and Co., New York (1972)
10. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press, Cambridge (2000)
11. Storn, R., Price, K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4), 341–359 (1997)
12. Tasoulis, D., Pavlidis, N., Plagianakos, V., Vrahatis, M.: Parallel differential evolution. In: IEEE Congress on Evolutionary Computation (2004)
13. Tuci, E., Ampatzis, C., Christensen, A.L., Trianni, V., Dorigo, M.: Self-assembly in physical autonomous robots: the evolutionary robotics approach. In: Proceedings of ALIFE XI, pp. 616–623. MIT Press, Cambridge (2008)

To Grip, or Not to Grip: Evolving Coordination in Autonomous Robots

Christos Ampatzis[1], Francisco C. Santos[2], Vito Trianni[3], and Elio Tuci[3]

[1] Advanced Concepts Team, European Space Agency, The Netherlands
Christos.Ampatzis@esa.int
[2] MLG & CoDE-IRIDIA, Université Libre de Bruxelles, Brussels, Belgium
[3] ISTC-CNR, Roma, Italy

Abstract. In evolutionary robotics, as in the animal world, performing a task which is beneficial to the entire group demands the coordination of different individuals. Whenever time-dependent dynamic allocation of roles is needed and individual roles are not pre-defined, coordination can often be hard to achieve. In this paper, we study the evolution of role allocation and self-assembling strategies in a group of two homogeneous robots. We show how robot coordination and individual choices (who will grip whom) can be successfully restated in terms of anti-coordination problems, showing how conventional game theoretical tools can be used in the interpretation and design of evolutionary outcomes in collective robotics. Moreover, we highlight and discuss striking similarities between the way our physical robots allocate roles and the way animals solve conflicts. Arguably, these similarities suggest that evolutionary robotics may offer apart from automatic controller design for autonomous robots a viable alternative for the study of biological phenomena.

Keywords: anti-coordination game, evolutionary robotics, collective behavior, evolutionary game theory.

1 Introduction

Robotics has largely drawn inspiration in the past decades from biology. Bio-inspired robotics refers to mimicking natural mechanisms at the hardware or the collective behaviour level. For example, social insects have often served as a source of inspiration for research on self-organized cooperative exploration in groups of robots using swarm intelligence techniques. Recently, the influence arrow that links biology to robotics has become bi-directional, as roboticists are implicitly or explicitly trying to answer questions related to biology, and in particular animal behaviour. This is because *"robots can be used as models of specific animal systems to test hypotheses regarding the control of behaviour"* [27].

Our view is that Evolutionary Robotics models (ER [17]) can be particularly suitable for testing hypotheses concerning both the **nature** and the **evolution** of the underlying mechanisms that underpin the agents' behaviour. Such models can be complementary to other analytical modelling tools at the disposal

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 205–212, 2011.

of biologists, such as (Evolutionary) Game Theory models (EGT, see [13], for example). While ER searches for the mechanisms to solve a problem given a fitness function, EGT defines high-level descriptions of any evolutionary process, useful for objectively identifying the conditions under which certain strategies can emerge as an evolutionary outcome. Recently, for example, ER models have contributed to the research on the evolution of communication and cooperation by proposing a low-level description and mechanisms at the neuronal level to realise signaling behaviour [12,9,2].

As a basis for our discussion we use the evolution of role allocation and self-assembling strategies in a group of two homogeneous robots (see [1,26]). The way in which we address the problem of having the agents coordinate to assume different roles (who will grip whom) is similar to anti-coordination problems. Coordination and anti-coordination problems are studied by biologists, either by direct observation of the behaviour of animals, or with the use of analytical modelling tools, as in [14], where "limited-war" type conflicts between conspecifics are studied. Game theory models allow biologists to predict the outcome of coordination/anti-coordination problems given the set of behavioural strategies available to the agent and the payoff corresponding to all the possible combinations of actions among the actors [13,10,22]. However, such analytical tools may be less suitable for testing hypotheses concerning the nature and the evolution of the underlying mechanisms that underpin the agents' behaviour. For example, as noticed by [3], we do not know the exact mechanisms (e.g., rules, signals and cues) involved in the formation of assembled structures in natural organisms. Should we assume that agents possess means for explicitly communicating each other's "intentions" in order to coordinate their actions? In order to address such questions, we believe that ER models can be suitable modelling tools, complementary to other analytical tools at the disposal of biologists.

Potential synergies between ER and EGT models are not limited there. We believe that the use of population dynamics and EGT provides a clear way of understanding (and designing) online adaptation and role allocation in populations of robots—central challenges in most collective robotics problems. This can provide a solid unified framework for the study of complex self-organized behavior in populations of robots.

2 Description of the Task and the Evolutionary Process

The task considered is described in detail in [1,26], and can be roughly summarised as follows:

- Two homogeneous robots are initialised at a random distance between a lower and an upper bound with random initial orientations;
- The robots do not have means for explicit communication;
- The agents should coordinate their movement in order to allocate roles and connect with each other via the gripping mechanism.

The controllers are evolved in a simulation environment which models some of the hardware characteristics of the *s-bot* (a small mobile autonomous robot with

self-assembling capabilities [15]). Each simulated *s-bot* is provided with an omni-directional camera mounted on its turret (see figure 1), returning the distance to the other *s-bot* in each of eight 45° sectors, up to a distance of 50 cm. The actuators controlled by the neural controller are the two wheels and the gripper actuator (open/close gripper). Artificial evolution was used to train networks which are cloned on two homogeneous robots. The network used is a Continuous Time Recurrent Neural Network (CTRNN [5]), with a feed-forward architecture, and the evolutionary algorithm used to set the parameters of the networks is the (μ,λ) evolutionary strategy ((μ,λ)-ES).

The fitness assigned to each genotype after evaluation of the robots behaviour is the average of the fitness achieved in 40 trials with predefined robot initialisations in order to ensure that successful controllers can cope with a large and representative sample of all possible initialisations. Notice that this set comprises both symmetrical and asymmetrical initial conditions, that is, when robots share the same initial perceptions, or not, respectively. In each trial, a group is rewarded by the following evaluation function, which assesses the ability of two robots to approach each other and assemble through the gripper, without dictating the role allocation strategy the robot should use:

$$\mathbf{F} = \mathbf{A} \cdot \mathbf{C} \cdot \mathbf{S} \tag{1}$$

A is the aggregation component used for bootstrapping purposes, computed as follows (with d the robot-robot distance at the end of the trial):

$$A = \begin{cases} \frac{1.0}{1.0+atan(\frac{d-16}{16})} & \text{if } d > 16 \text{ cm;} \\ 1.0 & \text{otherwise;} \end{cases} \tag{2}$$

C is the collision component aiming to gradually and smoothly punish collisions (with n the number of robot-robot collisions), computed as follows:

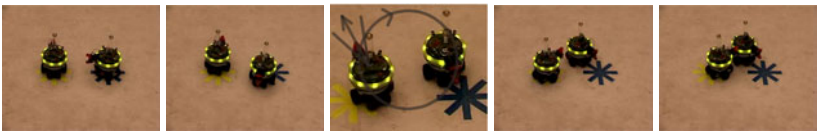
$$C = \begin{cases} 1.0 & \text{if } n = 0; \\ 0.0 & \text{if } n > 20; \\ \frac{1.0}{0.5+\sqrt{n}} & \text{otherwise;} \end{cases} \tag{3}$$

S is the self-assembly component, computed at the end of a trial as follows (with $K(t)$ a bootstrapping component):

$$S = \begin{cases} 100.0 & \text{if robots are assembled;} \\ 1.0 + K(t) & \text{otherwise;} \end{cases} \tag{4}$$

3 Successful Strategies Tested on Real Robots

One of the evolved neurocontrollers was first extensively tested in simulation under varying noise conditions to ensure its robustness and its generalisation capabilities with respect to various initial conditions. Subsequently, this controller



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

Fig. 1. Snapshots from a successful trial. (a) Initial configuration. (b) Starting phase. (c) Role allocation phase. (d) Gripping phase. (e) Success (grip).

was downloaded to two *s-bots*, and tested against different environmental conditions, with a very high success rate (see [1,26]).[1]

For all trials, histories of interactions can be described by transitions between a few phases which exhaustively "portray" the observed phenomena. The robots leave their respective starting positions (see figure 1a) and during the starting phase (see figure 1b) they approach each other. The robots then move from the starting phase to what we call the role allocation phase (RA-phase, see figure 1c), in which each *s-bot* tends to remain on the right side of the other. They slowly move by following a trajectory corresponding to an imaginary circle centred in between the *s-bots*. Moreover, each robot rhythmically changes its heading by turning left and right. The RA-phase ends once a robot (*s-bot-gripper*) stops oscillating and heads towards the other (*s-bot-grippee*), which instead orients itself to facilitate the gripping (gripping phase, see figure 1d). The *s-bot-gripper* approaches the *s-bot-grippee*'s turret and grips it. A successful trial terminates when the robots are connected (see figure 1e).

4 Self-assembly as an Anti-coordination Game

The results of post-evaluation analyses illustrate that the role allocation is the result of an autonomous negotiation phase between the robots [1,26]. The outcome of any action an agent chooses depends on the action of the other and none of the two agents can predict its final role from its initial perception. According to [11], in coordination and anti-coordination problems, *"two (or more) agents must choose one of several alternative actions"*. The author continues by stressing that *"the outcome of any action an agent might choose depends on the action of the other agents"*, as in every frequency dependent process [18]. In [1,26] it is shown that the system is characterised by two basic operational principles:

- For almost all asymmetrical initial configurations, the experimenter can predict the result of the role allocation;
- For all symmetrical initial configurations, we cannot predict who will grip whom and the agents assume both roles with 50% probability, as random noise in sensors/actuators is the element that breaks the symmetry.

[1] Videos available at `http://iridia.ulb.ac.be/supp/IridiaSupp2008-002`

This kind of scenario can be described as an anti-coordination problem, in which the agents have to estimate the strategy of their opponent. If the agents had the means to directly and precisely access the other's strategies, the role allocation would be immediate. However, our agents are deprived of any such information and must interact to assess and estimate each other's strategy.

The robots manage to coordinate their actions, "hovering" around the conditions that lead to assuming the *s-bot-gripper* role (oscillatory movement). The way the robots solve the anti-coordination problem bears striking similarities to how animals solve conflicts. In fact, the conflict between two strategies as attack and flee (or surrender), approach or avoid, is very common in nature. For example, gulls during fights adopt in turns agonistic (aggressive) postures which are abandoned as the birds turn broadside to the antagonist. Eventually, one bird will abandon the offensive and will adopt an appeasement posture or run away [25]. The similarity of this behaviour with the one of our robots is striking: it has even been observed that fighting birds walk parallel or around each other, as our robots circle around each other (see figure 1c). Similar coordination rituals are observed in gulls mating, in the "dance-fighting" observed in the male starling [7], in the fighting behaviour but also the mating "zig-zag dance" of the stickleback [24], and in the parallel walks engaged in by red deer stags [6], which allow for each animal to assess the other's size and strength and to investigate possible asymmetries [16]. Parallels can also be drawn between our system and simultaneous hermaphrodites, as snails, slugs and fish species, where individuals take single mating decisions ("one-shot" games) that require anti-coordination, since assuming the same role will end up costly for both [4].

Obviously, the way in which our robots solve the self-assembly task is determined by the way in which we set up our evolutionary process. By isolating the particular part of the fitness function related with the gripping decision, one immediately obtains a payoff matrix of a simple anti-coordination game, i.e., a simple version of the "Hawk-Dove" game[2] [13,14]. The fitness function selectively rewards the robot group to achieve self-assembly: the robots must coordinate to decide who will grip whom. A failure to take a decision will result in low fitness scores. The same goes if both robots decide to assume the *s-bot-gripper* role; the robots will collide and will be therefore punished by the fitness function. If we make a simplification and we assume that i) correct role allocation yields the maximum fitness (100), ii) a failure in the decision-making (with both robots playing *s-bot-gripper*) leads to a fitness score of 0, and iii) in case of failure to allocate the roles during the length of a trial, robots receive just the aggregation fitness component ($A \ll 1$), the payoff matrix would be the one in table 1.

Let's assume that each controller defines an abstract behaviour p, which in turn defines the probability of choosing to grip or not to grip. From table 1, it is trivial to see that $p = 1/2$ is the only evolutionary outcome of this system, whenever $A \to 0$. In other words, this problem has only one solution (the action

[2] This game is based on the principle that the outcome where neither player yields to the other is the worst possible one for both players.

Table 1. The payoff matrix of the game our robots are evolved to play. One robot chooses a strategy from the columns, and the other from the rows. The payoff refers to the fitness score assigned to the group after the end of the trial. A is the aggregation component of the fitness function (see equation 2).

	s-bot-gripper	*s-bot-grippee*
s-bot-gripper	0	100
s-bot-grippee	100	A

of every agent is optimal according to what the other agent does): that the agents should do the opposite of what the other is doing and thus allocate roles. More specifically, the circular movement with oscillations can be seen as the sum of two components: assuming the *s-bot-gripper* role and abandoning it. A premature decision on behalf of one robot to assume the *s-bot-gripper* role might lead to a decision-making error and the robots would end up receiving a fitness score of 0. Thus, a robot has to assume this role while the other assumes the *s-bot-grippee* role. This solution is optimal regardless of the selection type (group or individual); even if we were using heterogeneous pairs of robots which were not evaluated collectively but individually, this solution would still be optimal.

Describing the fitness function in game theory terms offers us the opportunity to view the experiment from a more high-level point of view and to realise that our experimental setup is very close to an anti-coordination game. This clarifies the outcome selected by evolution and shows that the solution found is not just a random one in a possible universe; instead, the principles characterising this solution could only be the ones they are. Also, this way of looking at our fitness function after understanding the basic mechanisms underpinning behaviour in our two robot system helps by providing a more high-level description of our system, that sometimes may be obscured by the complexity involved in the effort to break down its behaviour to a set of transparent rules or states.

Furthermore, even if it provides an excellent starting point, it is important to note that the complexity offered by the above one-shot analysis fails to embrace the complexity of the online decision process. The reason is twofold: first, robots do not play mixed strategies, but, instead, reactions to environmental inputs are preassigned by evolution; second, the online decision process and continuous integration of inputs from the environment, implies not a one-shot game, but a repeated anti-coordination game, in which more complex strategies can emerge. This is particularly relevant in the analysis of the case of symmetric initial configurations. In these cases, robots negotiate on-the-fly their strategies, as they have the possibility to wait in order to assess the other's decision and, at the same time, reshape their own strategy. In order to ease the differentiation and facilitate the convergence to complementary roles, evolution produced a new strategy characterized by the "dancing" movement (RA-phase, see figure 1c) which amplifies the effect of noise as a symmetry breaking factor. As expected, whenever the same decision process is studied in the absence of noise (in simulations, not shown) the RA-phase continues indefinitely.

5 Conclusions

We have introduced the idea of the feasibility of ER models as models for biological behaviour, complementary to game theory models. Using the game theoretical framework to design the rules of the game, the properties of the interaction space between agents and the costs and benefits they may share, we can practically build a fitness function within the ER framework incorporating these rules. This can be particularly beneficial when identifying the relative importance that should be given to different fitness components in order to obtain a systematic evolution of the desired behaviour. Subsequently, artificial evolution can propose bias-free low-level bechavioural mechanisms instead of high-level strategies to achieve the solutions to the "game" specified.

In this manuscript, only pairwise interactions were considered. However, both engineering and biological applications are often characterised by more than two individuals engaging in coordination and co-existence dilemmas [19]. In this case, allocation of roles and online social dynamics becomes even more intricate. Factors such as selection pressure, interaction structure, population size and average number partners [20,19,8] are known to play decisive roles, making a game theoretical analysis necessary to understand and design efficient populations of controllers. Similarly, to achieve coordination of actions, information needs to be transmitted and properly interpreted. Recent advances in dynamics of signaling and information processing [21,23] are able to capture the most essential aspects by means of simplified models based in game theory. Hence, also here, a high level population-based perspective for ER may provide a unified framework for the study of the evolution of communication, filtering, integration and resultant action. This is an open field of research that can complement the already present momentum that ER research on the evolution of communication has generated [12,9,2], offering a novel way of doing robotics, inspired by biology and population dynamics and less biased by the experimenter.

References

1. Ampatzis, C., Tuci, E., Trianni, V., Christensen, A.L., Dorigo, M.: Evolving self-assembly in autonomous homogeneous robots: experiments with two physical robots. Artificial Life 15(4), 1–20 (2009)
2. Ampatzis, C., Tuci, E., Trianni, V., Dorigo, M.: Evolution of signaling in a multirobot system: Categorization and communication. Adaptive Behavior 16(1), 5–26 (2008)
3. Anderson, C., Theraulaz, G., Deneubourg, J.-L.: Self-assemblages in insect societies. Insectes Sociaux 49(2), 99–110 (2002)
4. Anthes, N., Putz, A., Michiels, N.: Gender conflicts, sex role preferences and sperm trading in hermaphrodites: a new framework. Animal Behaviour 72, 1–12 (2006)
5. Beer, R., Gallagher, J.: Evolving dynamical neural networks for adaptive behavior. Adaptive Behavior 1(1), 91–122 (1992)
6. Clutton-Brock, T., Albon, S.D., Gibson, R.M., Guiness, F.E.: The logical stag: adaptive aspects of fighting in red deer (*cervus elaphus l.*). Animal Behaviour 27, 211–225 (1979)

7. Ellis, C.-J.R.: Agonistic behaviour in the male starling. The Wilson Bulletin 78(2), 208–224 (1966)
8. Pacheco, J.M., Santos, F.C., Lenaerts, T.: Evolutionary dynamics of social dilemmas in structured heterogeneous populations. PNAS 103, 3490–3494 (2006)
9. Floreano, D., Mitri, S., Magnenat, S., Keller, L.: Evolutionary conditions for the emergence of communication in robots. Current Biology 17, 514–519 (2007)
10. Hofbauer, J., Sigmund, K.: Evolutionary Games and Population Dynamics. Cambridge University Press, Cambridge (1998)
11. Lewis, D.: Convention: A philosophical study. Harvard University Press, Cambridge (1969)
12. Lipson, H.: Evolutionary robotics: Emergence of communication. Current Biology 17, R330–R332 (2007)
13. Maynard-Smith, J.: Evolution and the theory of games. Cambridge University Press, Cambridge (1982)
14. Maynard-Smith, J., Price, G.: The logic of animal conflict. Nature 246(5427), 15–18 (1973)
15. Mondada, F., Pettinaro, G.C., Guignard, A., Kwee, I.V., Floreano, D., Deneubourg, J.-L., Nolfi, S., Gambardella, L.M., Dorigo, M.: Swarm-bot: A new distributed robotic concept. Autonomous Robots 17(2-3), 193–221 (2004)
16. Noble, J.: Talk is cheap: Evolved strategies for communication and action in asymmetrical animal contests. In: Proceedings of SAB 2000, pp. 481–490. MIT Press, Cambridge (2000)
17. Nolfi, S., Floreano, D.: Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines. MIT Press, Cambridge (2000)
18. Nowak, M.: Evolutionary Dynamics: Exploring the Equations of Life. Harvard University Press (2006)
19. Pacheco, J.M., Santos, F.C., Souza, M., Skyrms, B.: Evolutionary dynamics of collective action in n-person stag hunt dilemmas. Proceedings Royal Society B 276(1655), 315–321 (2009)
20. Santos, F.C., Santos, M.D., Pacheco, J.M.: Social diversity promotes cooperation in public goods games. Nature 454, 213–216 (2008)
21. Skyrms, B.: Evolution of the social contract. Cambridge University Press, Cambridge (1996)
22. Skyrms, B.: The Stag Hunt and the Evolution of Social Structure. Cambridge University Press, Cambridge (2003)
23. Skyrms, B.: Evolution of signalling systems with multiple senders and receivers. Phil. Trans. R. Soc. B 364(1518), 771–779 (2009)
24. Tinbergen, N.: The curious behavior of the stickleback. Scientific American 187, 22–26 (1952)
25. Tinbergen, N.: The herring gull's world. Collins, London (1953)
26. Tuci, E., Ampatzis, C., Christensen, A.L., Trianni, V., Dorigo, M.: Self-assembly in physical autonomous robots: the evolutionary robotics approach. In: Proceedings of ALIFE XI, pp. 616–623. MIT Press, Cambridge (2008)
27. Webb, B.: What does robotics offer animal behaviour? Animal Behaviour 60, 545–558 (2000)

Development of Abstract Categories in Embodied Agents

Giuseppe Morlino[1,2], Andrea Sterbini[2], and Stefano Nolfi[1]

[1] Institute of Cognitive Sciences and Technologies (CNR-ISTC), Roma, Italy
[2] Department of Computer Science, Sapienza University, Roma, Italy
giuseppe.morlino@istc.cnr.it, sterbini@di.uniroma1.it,
stefano.nolfi@istc.cnr.it

Abstract. In this paper we demonstrate how a neuro-robot situated in an environment containing parallelepiped objects that vary in shape, size, and orientation can develop an ability to recognize and label the category of the objects and generalize to new objects. The analysis of the dynamical system constituted by the robot and the environment in interaction allowed us to understand how adapted agents solve the categorization problem at the level of the detailed mechanisms and at the level of the general strategy.

Keywords: Categorization, dynamical systems, evolutionary robotics.

1 Introduction

In this paper we demonstrate how a simulated neuro-robot situated in an environment containing parallelepiped objects that vary in shape, size, and orientation can develop through an evolutionary method [1] an ability to recognize and label the category of the objects (i.e. to discriminate whether the objects have a square or rectangular base). Since the sensors of the robots only provide information about a limited portion of the object, the categorization processes requires an ability to integrate sensory-motor information over time.

The goal of the paper is to study how a robot can associate sensory-motor values which vary continuously in state and time to abstract categories such us *square* or *rectangle* (a capacity that might represent a prerequisites for the development of several cognitive skills such us language [2]). Since the aim of the paper is not to investigate the role of active perception (i.e. how the possibility to co-determine the experienced stimuli through actions can be exploited to enable or to simplify the categorization process, c.f. [3-6]), the ability of the robot to explore the objects by traveling around them and to label their category have been evolved in two successive phases. The analysis of the coupled robot/environment system, also through the use of mathematical tools of dynamical system theory, allowed us to understand how the adapted agents solve the categorization problem both at the level of the detailed mechanisms and at the level of the general strategy. Moreover, the analysis conduced allowed us to elucidate the relation between the solution developed by the robots and the solution that might appear intuitive from the point of view of a human external observer that consists in measuring and comparing the length of two adjacent sides of the object and that involves a form of relational categorization, c.f. [7].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 213–221, 2011.

2 The Experimental Scenario

The experiment involves a Khepera robot [8] situated in an arena that contains a square or a rectangular parallelepiped (Fig. 1). The robot is provided with eight infrared proximity sensors (which detect obstacles up to a distance of ~4cm) and two motors which control the desired speed of the two corresponding wheels. The experiments have been carried out in simulation by accurately modelling the robot/environment interactions through a sampling technique [1].

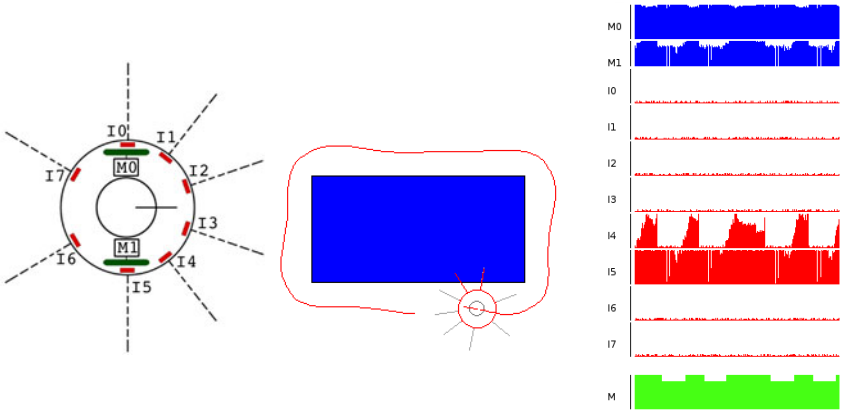


Fig. 1. Left: The position of the eight infrared sensors on the robot's body. Centre: The trajectory produced by an adapted robot that is traveling around a R object of 56x28 cm. Right: the activation of the left and right motors (M0-M1) controlling the two corresponding wheels and of the eight infrared sensors (I0-I7) during the behavior shown in the central picture. M is a neuron used in the second experiment that is set to 0.8 or 1.0 depending on whether M1 is below or above 0.9.

Each robot is evaluated for 40 trials during which it is allowed to interact with 20 square (S) and 20 rectangular (R) parallelepipeds of different sizes. The ratio between the length of the long and short sides in R objects is always ½. The depth and the width of the objects vary for each trial within [20, 80] cm and are selected so to ensure that each side length occurs with the same probability in the S and R objects during the 40 trials. At the beginning of each trial, the robot is positioned at the center of the south side of the object, oriented towards west, and the state of the internal neurons of the robot (see below) is set to 0.0. Each trial lasts 1000 time-steps and each step lasts 100ms.

The robot's control system consists of a neural network composed by two modules: a motor module (Fig. 2, left) that regulates the speed of the two wheels, and a categorization module (Fig. 2, right) that determines the robot's categorization output (label).

The motor module is composed by eight infrared sensory neurons (I0-I7) directly connected to two motor neurons (M0-M1). The categorization module is composed by two neurons (M0-M1$_{(t-1)}$ that encode the state of the two corresponding motor

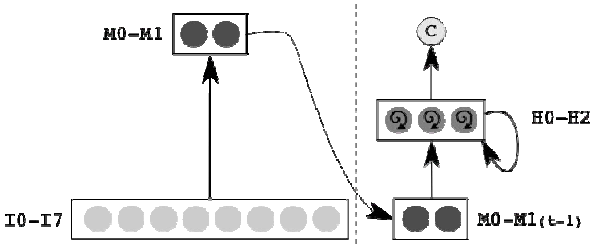


Fig. 2. The architecture of the robot's neural controller. The arrows between blocks indicate that all neurons of the second block receive connections from all neurons of the first block (or of the same block in the case of the three internal neurons).

neurons at time t-1), three internal neurons with recurrent connections (H0-H2), and one categorization output neuron (C).

The desired speed of the two wheels is set on the basis of the output of neurons M0-M1 normalized in the range [-10, 10]. The sensory neurons (I0-I7) are relay units which encode the state of the corresponding infrared sensors normalized in the range [0.0, 1.0]. The neurons (M0-M1$_{(t-1)}$) encode the state of the motor neurons at time t-1. The motor neurons (M0-M1) and the categorization unit (C) are updated according to the standard logistic function. The sensors, the neural controller, and the motors are updated every 100ms (the time step duration). The three internal neurons are leaky integrators which are updated on the basis of the following equations:

$$O_j = O_j^{(t-1)}\tau_j + \left(1 + e^{-A_j}\right)^{-1}\left(1 - \tau_j\right) \qquad A_j = t_j + \sum_i w_{ij} O_i \tag{1}$$

With A_j being the activity of the jth neuron, t_j being the bias of the jth neuron, w_{ij} the weight of the incoming connections from the ith to the jth neuron, O_i the output of the ith neuron, $O_j^{(t-1)}$ the output of the jth neuron at the previous time step, τ_j the time constant of the jth neuron.

The characteristics of the robot's body and of the architecture of the robot's neural controller are fixed. The connection weights, the biases, and the time constants of the internal neurons are adapted through an incremental evolutionary method (Nolfi & Floreano, 2000) that includes two phases.

The initial population consists of 100 randomly generated genotypes which encode the free parameters of 100 corresponding individuals. Each parameter is coded with 8-bit and is normalized in the interval [−5.0, +5.0] for the biases and the synaptic weights, and in the interval [0.0, 1.0] for the time constants. Each subsequent population is obtained by selecting and retaining the best 20 individuals (the *èlite*) of the previous population and by applying mutations (with 3% probability of flipping a bit) to 4 copies of each best individual.

During the first phase of the evolutionary process the free parameters of the motor neural module has been adapted for the ability to circumnavigate the object. More precisely, the fitness of the individuals has been calculated by computing the number or times the robot approaches a new corner of the object (i.e. every time the robot navigates from one corner to the next). This phase has been continued for 100 generations

during which the robot develop an ability to circumnavigate objects of different size by displaying a wall following behavior (see Fig. 1, centre).

During the second phase the free parameters of the motor neural module are fixed on the basis of the parameters of the best individual obtained during the previous phase while the parameters of the categorization neural module have been evolved for the ability to label the category of the object. More precisely, the fitness of the individuals consists of the average absolute difference between the output of the categorization unit (C) and the desired value (i.e. 0.0 and 1.0 for S and R objects, respectively) calculated in each time step during the second half of each trial. The second phase is continued for 2000 generations and replicated 20 times starting from different randomly initialized genotypes.

2 Results

The analysis of the fitness at the end of the evolutionary process indicates that the evolved robots display close to optimal performance (more than 95% of correct categorizations) in 3 out of 20 replications, and good but sub-optimal performance in the other replications of the experiment. Moreover, we observed that the best adapted individuals display a remarkable ability to generalize their skill (within limits) to objects that differ, respect to those experienced during the adaptive process, either in size and/or in the ratio between their shorter and the longer sides.

Fig. 3 (left), shows the results of a test in which the best adapted robot is evaluated for 10,000 trials in a test condition in which we systematically varied the length of the north/south and east/west side within [10, 200] cm. As shown in the figure, in fact, the robot categorizes correctly S and R objects in the range [20, 100] cm.

The analysis also demonstrates that the adapted robot displays the two constituting properties of categorical perception: *labelling*, i.e. the capacity to partition stimuli varying in a continuous manner into well distinct classes, and *discrimination*, i.e. the tendency to better distinguishing between classes than within classes [5, 9-10]. More specifically, the presence of sharp boundaries between the two categories demonstrates that the robot partitions objects varying in a continuous manner into two well differentiated categories. As can be seen in the figure, objects are partitioned between the two classes on the basis of whether the ratio between their shorter and longer sides is higher or lower that ~0.7 in a way that is substantially independent from the size of the objects within the [20, 100] cm range. The sharp transition between the two categories shows that the output of the categorization unit (C) varies more for objects of different categories than for objects of the same category. These labelling and discrimination properties show that the robot categorizes objects (with side/ratios that differ from those experienced during the adaptive process) on the basis of the similarity between their sides/ratio and the sides/ratio of the S and R objects experienced during the adaptive process. In other words the analysis shows that the robot generalizes its skill also for objects with different sides/ratio. Moreover, the symmetry of the figure with respect to L1 and L2, that represent the length of the north/south and east/west sides, demonstrates that the robot categorizes rectangular objects correctly independently from whether they are oriented vertically or horizontally.

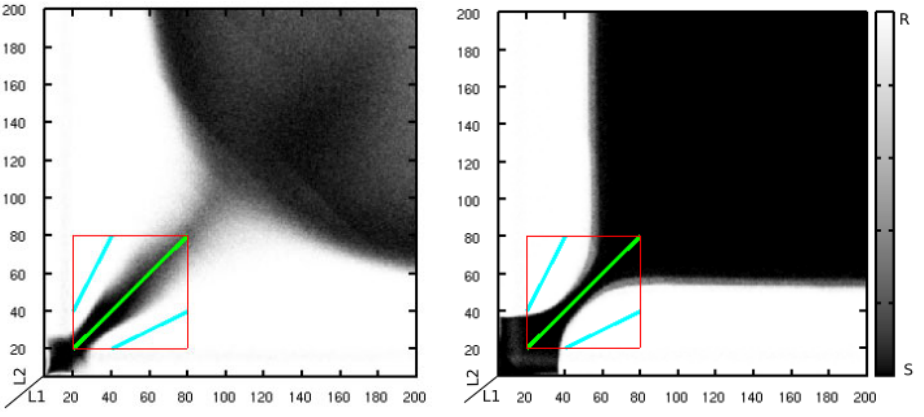


Fig. 3. Categorization outputs produced in interaction with objects with sides varying within [10, 200] cm. The L1 and L2 axis represents the length of the east/west and north/south sides of the objects, respectively. The greyscale colour represents the value of the categorization unit at the last time step averaged over 10,000 trials (0.0 = black, 1.0 = white). The central and the other two shorter lines inside the bottom-left square represent the size variation of the S and R objects experienced during adaptive process. Left: Results for the best individual of the basic experiment. Right: Results for the best individual of the simplified experiment (see section 2.1).

2.1 Dynamical Analysis of the Coupled Robot-Environmental System

To understand how the evolved robots categorize the two classes of objects we analysed the coupled dynamical system constituted by the robot and the environment. To overcome the problems due to the high dimensionality of the system we decided to analyse a slightly simplified version of the model. More specifically, in consideration of the fact that the two input neurons of the categorization module (M0-$M1_{(t-1)}$) encode redundant information, we ran a second experiment in which the categorization module includes only one neuron (M) whose activation state is set to 0.8 or 1.0 when the activation of the motor neuron M1 at time t-1 is lower or greater than 0.9, respectively (see Fig. 1, right).

The obtained results indicate that the simplified model leads to qualitatively and quantitatively similar results (see Fig.3, right). Fig. 4 displays the dynamics of the three internal neurons of the categorization module of the best adapted individual, produced when the state of the M neuron is fixed to 0.8 or 1.0 (left and right figures, respectively). Fig. 5 displays the typical trajectories of the same three internal neurons produced by the coupled dynamical system constituted by the robot and the environment in interaction.

One first thing to notice is that the state space includes four transient attractor points of which two (ACR/ASR) are located on the top and two (ACQ/ASQ) are located on the bottom of the state space. The attractors are transient since they manifest themselves when the M unit assumes one of the two possible values. More specifically, ACR and ACQ manifest themselves when the robot is negotiating a corner (i.e. when M=0.8) and ASR and ASQ manifest themselves when the robot is travelling along a side of the object (i.e. when M=1.0). The ACR and ASR attractors

are located nearby in the top part of the state space while the ACQ and ASQ attractors are well separated in the bottom part of the state space.

Secondly the top and the bottom part of the state space (with the exception of the small corner area near P0) trigger the R and S categorization answers, respectively.

The third thing to notice is that the basin of attractions of ACR, ACQ and ASQ are confined in their relative areas (i.e. in the top part of the state space for ACR and in the bottom part for ACQ and ASQ). The basin of attraction of ASR, instead, extends from the bottom to the top part of the state space and can thus bring the internal state of the robot from the bottom to the top part of the state space.

The fact that at the beginning of the trial the internal state starts from P0 and then move quickly toward the ASQ and ACQ transient attractors implies that, after few time steps, the robot starts to produce S as a default categorization answer. Moreover it implies that for squared objects the state of the internal neurons remains in the bottom part of the state space (from which an S categorization answer is produced) while for rectangular objects, at a certain point, the state moves along the ASR basin of attraction from the bottom to the top part of the state space (from which an R categorization answer is produced). The fact that the ASQ and ACQ basins of attraction are confined on the bottom part of the state space (i.e. the fact that they cannot bring the state of the internal neurons from the top to the bottom part of the state space) implies that the R categorization answer is irreversible.

Overall, these considerations imply that, to understand how categorization occurs we should understand the conditions that determine whether the state of the internal neurons remains in the bottom part of the state space or it moves and then remains on the top part of the state space. In other words the conditions that determine whether the robot keeps producing the default answer (S) or it starts producing the alternative answer (R).

When the robot travels along a side of the object, the state space is dominated by the ASQ and ASR attractor points. Since the state of the internal neurons is initially set to P0, however, during the initial phase of the trial the trajectory of the internal state is affected only by the basin of attraction of ASQ, that is located on the bottom part of the state space. As soon as the robot negotiates a corner of the object, the previous attractors are replaced by ACQ and ACR. Since the bottom part of the state space is dominated by the basin of attraction of ACQ, the state of the internal neurons then starts moving toward the ACQ attractor point. The periodic alternation of the two transient attractors thus leads to a stable or quasi-stable limit cycle, in which the state of the internal neurons move toward the ASQ and ACQ attractors located on two opposite sides of the bottom part of the state space. During the exhibition of this limit cycle the state of the internal neurons never fully reaches the two attractor points due to the limited time duration of each attractor, the inertial nature of the internal neurons, and the fact that the internal state moves more quickly toward the latter than toward the former attractor. Whether or not the basin of attraction of ASR succeeds in breaking this limit cycle dynamics and in bringing the internal state in the top part of the state space depends from the following factors.

The first factor concerns the fact that the position and the extension of the limit cycles that emerge from the robot/environmental interaction vary along the ACQ and ASQ dimension depending on the length of the last sides of the object negotiated by the robot.

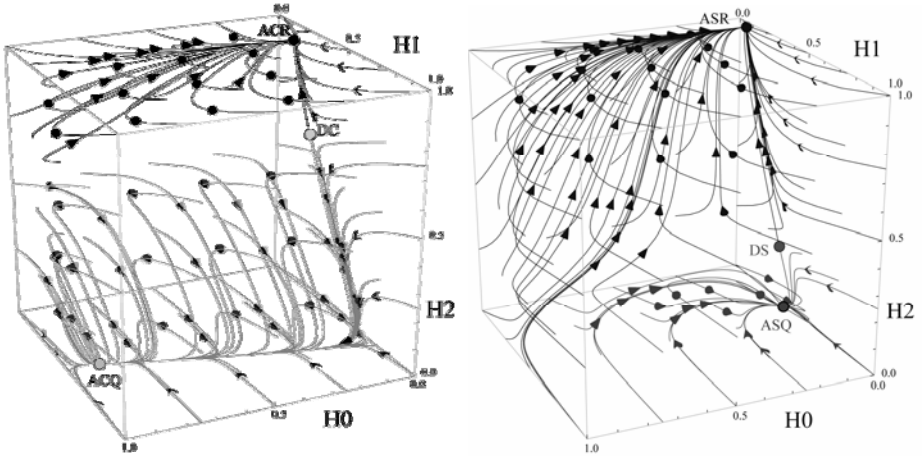


Fig. 4. Flow and phase portrait of the categorization module of the best adapted individual, produced when the state of M is fixed to 0.8 and 1.0 (left and right pictures respectively) that correspond to the states experienced by the robot when it travels along a corner or a side of the object, respectively. The three axes represent the state of the three internal neurons. The letter A stands for "attractor" (i.e. fixed point attractor), D for "saddle" (i.e. repellor), C for "corner", S for "side", Q for "cube/square" and R for "rectangle"; so for example ASR indicates the fixed attractor point that manifests itself when M=1.0 (when the robot is travelling along a side of the obejct) and that triggers a R answer.

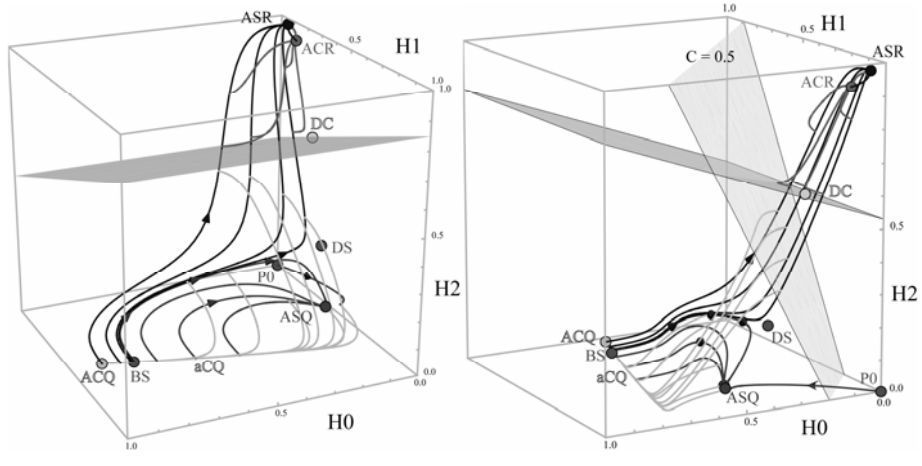


Fig. 5. Sample trajectories produced by the coupled dynamical system constituted by the robot and the environment in interaction. The left and right pictures display the same 3D structure from two different points of view. The three axes represent the state of the three internal neurons. The darker and lighter lines indicate the trajectory produced when the M neuron assumes a value of 0.8 or 1.0, respectively. P0 indicates the initial state of the internal neurons. The decision plane that intersects the DC saddle point indicates the border between the basins of attractions of the ACR and ACQ attractor points. In the right picture, the categorization plan indicated with C=0.5 separates the two areas or the state space that trigger an S or R categorization answers, respectively.

The extent to which the internal state approaches the ACQ attractor point primarily depends from the distance between the state of the internal neurons and the attractor at the beginning of the negotiation of the corner, which in turn is inversely proportional to the time duration of ASQ attractor that manifested itself while the robot was travelling along the previous side. We say "primarily" since the extension of the limit cycle toward the ACQ attractor also depends from the length of sides negotiated before the last one, thanks to the same effects described above for the last side (although the impact of previous stimuli tends to become progressively less important over time). The length of the last side negotiated by the robot (and, secondarily, the length of the sides negotiated before the last one) thus determines whether, while moving toward the ACQ attractor point, the internal state overcomes the BS point so to enter (after the negotiation of the corner) into the basin of attraction of the ASR attractor point located on the top part of the state space. The extent to which the internal state approaches the ACQ attractor also depends from the amount of time in which the attractor manifests itself that, however, is approximately the same for all corners (independently of whether they belong to square or rectangular objects).

The second factor that determines whether the state of the internal neurons crosses the decision plane that intersect the DC saddle point and enters (and then remains) into the top part of the state space or not, depends from the time duration of ASR (i.e. from the length of the current side) and from the extension of the limit cycle toward ACQ (that is inversely proportional to the length of the previous side, primarily, and of the sides before the previous, secondarily).

This type of analysis reveals also why the robot's generalization ability is limited within the range described earlier. Additional information on this point as well as additional explanatory material is available from http://laral.istc.cnr.it/esm/abstract-categorization/.

3 Conclusions

In this paper we demonstrated how a simulated neuro-robot can develop an ability to associate sensory-motor stimuli which vary continuously in state and time to abstract categories. The analysis of the coupled dynamical system constituted by the robot and the environment in interaction demonstrates that the problem is solved by exploiting dynamical processes occurring at different time scales and the fact that the stimuli experienced by the robot can act as parameters that lead to sharp transitions in the robots' internal dynamic. More specifically, the slow dynamic that originates from the inertial nature of the internal neurons allows the robot to detect and to keep trace in its internal state of the duration of previously experienced events (e.g. the time duration of the action produced by the robot along a side of the object). On the other hand, the sudden alternation of different type of stimuli lasting for certain time durations allows the robot to sudden rearrange its internal dynamic in crucial phases (e.g. to perform an implicit comparison between the length of current and previous events). Moreover, it is exploited to produce sharp transitions in the robot's internal dynamic (e.g. to channel the state of the internal neurons toward different areas of the state space associated to different categories).

At a more general level of description, the solution developed by the evolved robots demonstrates how the problem admits different solutions, including solutions that are more parsimonious and robust with respect to those that can be identified by an external observer. In particular, the problem faced by the robot is solved without fully partitioning the quantity to be compared (with particular reference to the length of the previous side) and by exploiting all available regularities (e.g. the overall size of the object and the fact that, in the domain of the experiment, very large and small objects always belong to the square category).

Acknowledgments. This research work was supported by the ITALK project (EU, ICT, Cognitive Systems and Robotics Integrating Project, grant n◦ 214668). The dynamical analysis as been carried on with Dynamica© [11]. The authors would also like to acknowledge Vito Trianni for his valuable insights on the analyses.

References

1. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press/Bradford Books, Cambridge, MA (2000)
2. Harnad, S.: To Cognize is to Categorize: Cognition is Categorization. In: Cohen, H., Lefebvre, C. (eds.) Handbook of Categorization in Cognitive Science. Elsevier, Amsterdam (2005)
3. Scheier, C., Pfeifer, R., Kunyioshi, Y.: Embedded neural networks: exploiting constraints. Neural Networks 11(7-8), 1551–1596 (1998)
4. Nolfi, S., Marocco, D.: Active perception: A sensorimotor account of object categorisation. In: Hallam, B., Floreano, D., Hallam, J., Hayes, G., Meyer, J.-A. (eds.) Proceedings of the 7th Inernational Conference on Simulation of Adaptive Behavior, pp. 266–271. MIT Press, Cambridge (2002)
5. Beer, R.D.: The dynamics of active categorical perception in an evolved model agent. Adaptive Behavior 11(4), 209–243 (2003)
6. Nolfi, S.: Categories formation in self-organizing embodied agents. In: Cohen, H., Lefebvre, C. (eds.) Handbook of Categorization in Cognitive Science, pp. 869–889. Elsevier, Amsterdam (2005)
7. Williams, P., Beer, R., Gasser, M.: An embodied dynamical approach to relational categorization. In: Love, B., McRae, K., Sloutsky, V. (eds.) Proceedings of the 30th Annual Conference of the Cognitive Science Society, pp. 223–228. Cognitive Science Society, Inc. (2008)
8. Mondada, F., Franzi, E., Ienne, P.: Mobile Robot miniaturisation: A tool for investigation in control algorithms. In: Proceedings of the Third International Symposium on Experimental Robotics, Kyoto, Japan (1993)
9. Studdert-Kennedy, M., Liberman, A.M., Harris, K.S., Cooper, F.S.: Motor theory of speech perception: A reply to Lane's critical review. Psychological Review 77, 234–249 (1970)
10. Ehret, G.: Categorical perception of sound signals: Factsand hypotheses from animal studies. In: Harnad, S. (ed.) Categorical Perception: The Groundwork of Cognition, pp. 301–331. Cambridge University Press, Cambridge (1987)
11. Beer, R.D.: Dynamica Tutorial. Version 1.0. First public release (2008)

For Corvids together Is Better
A Model of Cooperation in Evolutionary Robotics

Michela Ponticorvo[1], Orazio Miglino[1,2], and Onofrio Gigliotta[2]

[1] Natural and Artificial Cognition Laboratory,
University of Naples "Federico II", Italy
[2] Laboratory of Autonomous Robotics and Artificial Life, Institute of Cognitive Sciences and Technologies, National Research Council, Rome, Italy

Abstract. In this paper we describe a model of cooperation in evolutionary robotics (ER) derived by animal research on Corvids. In recent years many researchers have proposed models of ER which are bio-inspired. The main source of inspiration has come from social insects, such as ants. Inspiration may come also from other representatives in the animal kingdom such as primates or corvids, thus producing different models that can address different issues. The work presented here starts from works inspired by social insects and then describes an ER model inspired by tasks in corvids, that addresses the evolution of cooperation, showing how different bio-inspired models can be useful to study different issues.

1 Introduction

In the first years of its life, evolutionary robotics (ER), [1, 2, 3] the fruitful technique for creation of autonomous robots based on the mechanism of Darwinian evolution, focused on the emergence of quite simple behavior such as obstacle avoidance or garbage collection [4] and on definition of its methods and techniques, for example the "simulate-and transfer" method [5]. These efforts were meant to give a clear identity to this new-born discipline, which in subsequent years, was able to carve for itself an interesting and stimulating niche in the survey of scientific literature about robotics and its application to cognitive science.

After this infancy period, researchers in ER started to look for methods which could lead to more and more complex behaviors. At this point they could choose between two alternatives: augmenting complexity inside the robot or augmenting complexity outside the robot. These alternative are expressed effectively by Izquierdo-Torres at the University of Sussex [6]: "Nature has been able to evolve (several times) natural systems which produce complex spatio-temporal patterns from agents with very simple behaviors by exploiting the interactions between the agents and their environment (...). In social insects large numbers of simple agents collectively achieve remarkable feats through exploiting a few principles. They offer a spectacular existence proof of the possibility of using many simple agents rather than one or a few complex agents to perform complex tasks quickly

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 222–229, 2011.

and reliably". In other words, complex behaviors may result from one or two quite complex agents or from many very simple agents that interact with their environment and self-organize under evolutionary pressure. If we quickly review ER literature it seems quite evident that this second option overcame the first one. Collective robotics [7, 8] is nowadays a consolidated frame of reference which aims at building multi-agent system in which robots are able to accomplish certain tasks by coordination among autonomous agents. In a certain sense, skills are distributed over a colony that must cooperate and communicate (also indirectly, in this case we refer to stigmergy in biological literature). In collective robotics agents self-organize producing complex, apparently intelligent structures, without need for any planning, control, or direct communication between agents. The main metaphor used is the swarm (shoaling, swarming or flocking), a term that is applied to fish, insects, birds and microorganisms, such as bacteria, and describes a behavior of an aggregation of similar organisms in which group size is a relevant factor. In the swarm the single agent is not important, only the swarm itself is relevant. This metaphor applied to robotics generated the emergent field of swarm robotics [9, 10, 11, 12] that studies robotic systems composed of swarms of robots. The agents in the swarm are in close interact and cooperate to reach their goal, just like what happens in social insects. If we consider the phenomena of waggle dance of the honey bee, the nest-building of the social wasp and the construction of the termite mound, we must admit that is amazing that these seemingly uncommunicative, very simple creatures are able to manifest these behaviors. They are able to do this by relying on simple mechanisms that produce notable effects, such as the above cited stigmergy and self-organization. The swarm metaphor emphasizes the decentralization of the control, limited communication abilities among agents, use of local information, emergence of global behavior and robustness that are particularly consonant with ER principles. In a swarm robotic system, although each single robot of the swarm is fully autonomous, the swarm as a whole can solve problems that the single robot cannot cope with because of physical constraints or limited capabilities.

Social insects are undoubtedly a precious source of inspiration, but we should not restrict our attention on it, as challenging cues may derive from others representatives of the animal kingdom. In other word, following McFarland distinction [13], we should pay attention not only to eusocial behaviors (found in many insect species and resulting from genetically determined individual behavior) but also to cooperative behavior. In the case of cooperative behavior there are not many very simple agents that interact, but two or more quite complex individuals that work together to reach a goal that would be otherwise impossible to obtain. In nature there are many cases of this kind of cooperation: for example we can observe coalitions (help provided during conflicts) and alliances (long-term association) in many animals: primates (chimps, baboons, macaques, vervets, capuchins), carnivores (lions, cheetahs, hyenas) and dolphins. Between other animals coalitions and alliances have been described also in corvids [14]. In this case what we observe is a small number of corvids, each of which is capable of refined cognitive abilities, that cooperate. They are an example of the first

alternative we described, increased complexity inside the agent, that has not been exploited as much as the second one.

In the next section, we propose a basic simulation based on cooperation tasks in corvids that will allow us to discuss the importance of different kind of cooperative models in ER.

1.1 Cooperation in Corvids

Corvids (*Corvidae*) family include various birds species characterized by high complexity in cognitive functions: they can be compared with primates both on brain relative dimensions, cognitive abilities and on social organization complexity [15, 16, 17]. They are capable of long term cache recovery, object permanence [18], tool manipulation, theory of mind like-abilities [19] and social reasoning. In nature we can observe them in dyads as well as in small or large colonies. Corvids are also able to cooperate in order to obtain a goal [20]. In the present study we propose a model that replicates in the main aspects the "loose string" paradigm derived from the Game Theory, applied to comparative research. In the "loose string" task two agents, for example two rooks (*Corvus frugilegus*), must cooperate to obtain a reward, i.e. food, which is clearly visible, but not directly reachable. The dyad gets the reward if the two tips of a string are pulled at the same time. In the present study we model this task with artificial organisms to study cooperation in artificial organisms.

In cooperation it is crucial to distinguish if dyads are "coordinated trough communication or acting apart together" [21] It seems therefore quite relevant trying to understand if communication allows dyads to cooperate indeed.

1.2 The "Loose String" Task

In the "loose string" task two members of a dyad are trained to pull a string to reach a reward. In a first phase, the agents, for example, corvids such as rooks [20], are trained separately to pull the string which allows the bird the get the food by itself. In the cooperation testing phase, the two birds could get the reward only if they pulled the string at the same time (see Fig. 1). In this task the members of the dyad exchange signals mainly on the visual channel (private communication) thus indicating each other where they are. We reproduced this natural experimental task with simulated robots.

2 Materials and Method

2.1 The Experimental Set-Up

The experimental setup involves two robots situated in a rectangular arena (1200 cm * 800 cm). Robots begin each trial at one end of the arena. On the other hand there are two target areas which robots must reach at about the same time. This task represents a situation in which the robots should coordinate themselves/cooperate to get a reward.

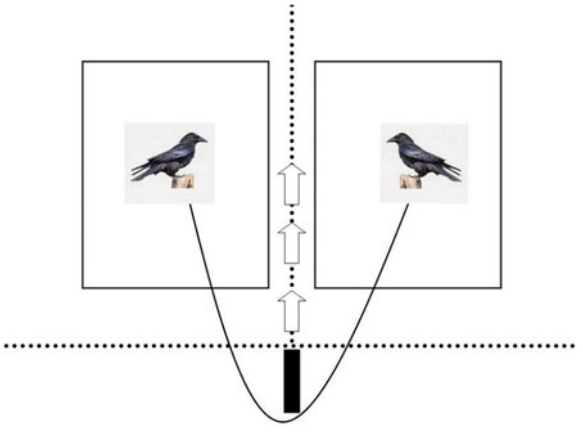


Fig. 1. The loose string task: two rooks must pull the string together to get the reward

To verify if the task required coordination or could be solved just acting apart together [21] we have run two simulations: in the first one robots could exchange signals on their relative position (see next section), in the second one they didn't communicate at all.

2.2 The Robot and Its Artificial Neural Controller

The robots are two e-Puck robots [22], with a diameter of 7.5 cm provided with 2 motors which control the 2 corresponding wheels, 8 infrared proximity sensors located around the robot's body, a ground sensor and a turret to send and receive signals on distance and angle of the other robot. The neural controller of each robot is provided with sensory neurons, internal neurons with recurrent connections and motor neurons. These neurons allow to receive and produce signals that can be perceived by another robot. In detail in the sensory layer there are neurons that encode activation of infrared sensors, ground sensor and distance/angle sensors; in the hidden layers there are 4 hidden neurons with recurrent connections; in the output layer there are two units that control wheels.

2.3 The Evolutionary Algorithm

An evolutionary technique is used to set the weights of the robots' neural controller. The initial population consists of 100 randomly generated genotypes that encode the connection weights of 100 corresponding artificial neural networks. Each genotype is translated into 2 neural controllers which are transferred in 2 corresponding simulated robots. The 20 best genotypes of each generation are allowed to reproduce by generating 5 copies each, with 2 % of their bits replaced with a new randomly selected value. The evolutionary process lasts 100 generations (i.e. the process of testing, selecting and reproducing robots is iterated 100

times). The experiment is replicated 10 times each consisting of 4 trials (1000 cycles each) with different starting direction face on one hand of the arena.

We used the following fitness function to evolve robots: if both robots are on the target areas they are rewarded. This reward corresponds to (minus) time lapse between reaching the target area by the first robot and by the second one. We thus reward reaching the target area at the same time.

3 Results

3.1 Fitness Values

In this section we compare the fitness values for two experimental conditions: with and without signals exchange. For each condition there are 10 replications with different seeds.

We compare the fitness value gained by the best robot of the last generation for each seed. The mean value for "with" condition is higher than in "without" condition: 97.16 (s.d. 18.23) versus 69.60 (s.d. 28.78); this difference was statistically significant: t test (9) = 2.83, p = 0.019.

Moreover, as the standard deviation is lower in "with" condition, the presence of signals allows better and comparable results with different starting conditions.

3.2 Behavioural Analysis

In this section we describe two prototypical dyads, one "with" and one "without" signals accomplish the task.

Their trajectories are depicted in Figures 2 and 3. The dyad "with" is able to coordinate: they wait for each other at the starting position, adjust their face-direction and go straight-on together up to the target areas. In the dyad "without", on the contrary, each robot gets to the target area on its own, thus the time lapse is higher. In other word, in presence of signals the robots are

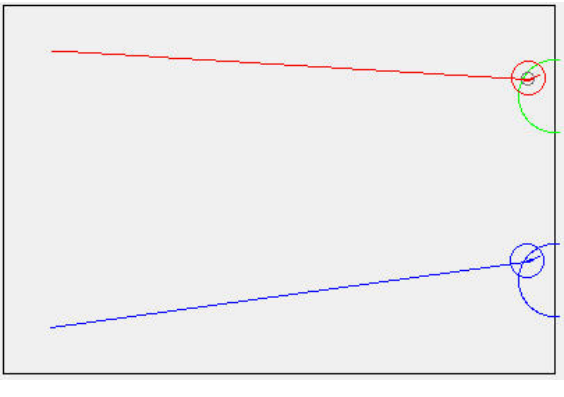


Fig. 2. Trajectories of the dyad "with" signals

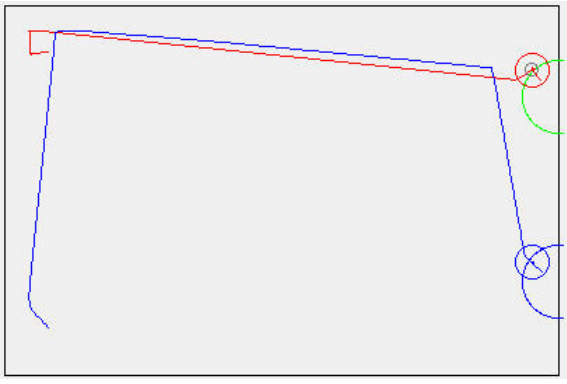


Fig. 3. Trajectories of the dyad "without" signals

able to use this primitive form of communication ("I know where you are") to solve efficiently the task, while in absence of signal exchange neither this simple cooperative behaviour can emerge.

4 Discussion and Conclusions

Results show that cooperation between robots is regulated by interaction between robots, with communication/signalling as a medium. In our simulative scenario the communication leads to a coordinated cooperation behavior in the task solved by natural organisms such as corvids.

What we would like to suggest with the simple experiment described above is to establish a strong link with phenomena and tasks derived from experiments on animal behavior in order to get insight from this kind of data reciprocally. For this reason we modeled a well-defined experimental set-ups, that has been widely used in animal behavior literature and try to compare what happens in corvids' cooperation with what happens in robots' cooperation. This exchange can be fruitful both for researchers working with natural organisms and for researchers working with artificial organisms. One of the hint for artificial-lifers may be the following: it is worth modeling more complex forms of cooperation. Different models of cooperation, in fact, may be used to study different issues: the first kind of models we introduced, with many simple agents, is useful to study self-organization, stigmergy and other issues related to biology. The second kind of models allows us to study, for example, how signals can be used, how the other agent's position is represented, in other words issues that are more related to psychology and cognition.

In this respect the present experiment is similar to the one described by Marocco and Nolfi [23]. In that paper authors describe how a population of simulated robots evolved for the ability to solve a collective navigation problem. In order to achieve this they evolve communication skills. What is different in our study is the theoretical focus: our experiment is just a simple example

to underline that taking inspiration from complex organisms may be a fruitful scientific strategy.

In experiments about rooks, for example, authors talk about personality, a concept that cannot be dealt with swarm metaphor. Nonetheless understanding these problems, even if very difficult to face also with the models we have introduced above, could receive beneficial if modeling is inspired by complex natural occurring forms of cooperation, such as grooming in primates or human cooperation. More complex cooperation modeling should couple with insect metaphor, which is so powerful for many reasons. First of all, modeling insects' eusocial behavior is favorite because insects are much more similar to evolved robots in ER if compared with other animals. A simulated or physical e-puck robot can be imagined as an ant, for example, much more easily than as a primate. Moreover, this kind of modeling permits to address problems that represent nowadays the heart of research in robotics in general and in collective robotics in particular, such as dynamical systems, distributed control, embodiment and situatedness, adaptive systems, coordination between autonomous agents, self-organization, etc. On the other side, trying to model agents or robots that are complex inside, would require a strong investment in cognitive science and from cognitive science that doesn't seem to be as strong as the previous one. The principal drawback in this kind of modeling is that, with actual techniques, it is still too difficult to build an agent that resembles in complexity natural organisms and this can be discouraging. One may object, for example, that the agents we used in our simulation are not at all comparable with corvids. This is undoubtedly true. In spite of this, this kind of bio-inspired models may be the first step in direction of modeling cooperation between complex agents: cooperation modeling may deepen our understanding of cooperation in group-living organisms and understanding cooperation in group-living organisms may allow to better understand how to build efficient artificial organisms.

References

1. Harvey, I., Di Paolo, E., Tuci, E., Wood, R., Quinn, M.: Evolutionary Robotics: A new scientific tool for studying cognition. Artificial Life 11(1/2), 79–98 (2005)
2. Harvey, I., Husbands, P., Cliff, D., Thompson, A., Jakobi, N.: Evolutionary Robotics: the Sussex Approach. Robotics and Autonomous Systems 20, 205–224 (1997)
3. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press/Bradford Books, Cambridge, MA (2000)
4. Nolfi, S., Floreano, D., Miglino, O., Mondada, F.: How to evolve autonomous robots: Different approaches in evolutionary robotics. In: Conf. Artificial Life IV (1994)
5. Miglino, O., Lund, H.H., Nolfi, S.: Evolving mobile robots in simulated and real environments. Artificial Life 2(4), 417–434 (1995)
6. Izquierdo-Torres, E.: Collective intelligence in multi-agent robotics: Stigmergy, self-organization and evolution. University of Sussex (2004)

7. Kube, C.R., Zhang, H.: Collective robotics: From social insects to robots. Adaptive Behavior 2(2), 189–219 (1993)
8. Beckers, R., Holland, O.E., Deneubourg, J.L.: From local actions to global tasks: Stigmergy and collective robotics. In: Proc. A-Life IV. MIT Press, Cambridge (1994)
9. Beni, G., Wang, J.: Swarm intelligence. In: Proceedings of the Seventh Annual Meeting of the Robotics Society of Japan, pp. 425–428. RSJ Press, Tokio (1989)
10. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York (1999)
11. Cao, Y.U., Fukunaga, A.S., Kahng, A.B.: Cooperative mobile robotics: Antecedents and directions. Autonomous Robots 4, 1–23 (1997)
12. Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chretien, L.: The dynamics of collective sorting: Robot-like ant and ant-like robot. In: Meyer, J.-A., Wilson, S.W. (eds.) From Animals to Animats. Proceedings of the First International Conference on Simulation of Adaptive Behavior (SAB 1990), pp. 356–365. MIT Press, Cambridge (1990)
13. McFarland, D.: Towards robot cooperation. In: Proc. Simulation of Adaptive Behavior (1994)
14. Clayton, N.S., Emery, N.J.: The social life of corvids. Current Biology 17, R652–R656 (2007)
15. Emery, N.J.: Are corvids 'feathered apes'? Cognitive evolution in crows, jays, rooks and jackdaws. In: Watanabe, S. (ed.) Comparative analysis of minds. Keio University Press, Tokyo (2004)
16. Emery, N.J., Clayton, N.S.: Comparing the complex cognition of Birds and Primates. In: Rogers, L.J., Kaplan, G. (eds.) Comparative vertebrate cognition: are primates superior to non-primates?, pp. 3–46. Kluwer Academic/Plenum, New York (2004)
17. Emery, N.J., Clayton, N.S.: The mentality of crows: convergent evolution of intelligence in corvid and apes. Science 306, 1903–1907 (2004)
18. Zucca, P., Milos, N., Vallortigara, G.: Piagetian object permanence and its development in Eurasian Jays (Garrulus glandarius). Animal Cognition 10, 243–258 (2007)
19. Bugnyar, T.: An integrative approach to the study of ToM-like abilities in ravens. Japanese Journal of Animal Psychology 57, 15–27 (2007)
20. Scheid, C., Noe, R.: Implications of individual temperament in a cooperative task (submitted)
21. Noe, R.: Cooperation experiments: coordination through communication versus acting apart together. Animal Behaviour 71, 1–18 (2006)
22. Mondada, F., Bonani, M.: The e-puck education robot (2007), http://www.e-puck.org/
23. Marocco, D., Nolfi, S.: Origins of communication in evolving robots. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J., Marocco, D., Miglino, O., Meyer, J.-A., Parisi, D. (eds.) SAB 2006. LNCS (LNAI), vol. 4095, pp. 789–803. Springer, Heidelberg (2006)

Dynamical Systems Analysis of a Protocell Lipid Compartment

Ben Shirt-Ediss

Evolutionary and Adaptive Systems, University of Sussex, Brighton, BN1 9QH, UK
ben@shirt-ediss.me

Abstract. This paper develops phase portraits explaining the dynamic behaviour of the lipid compartment in a recently proposed stochastic protocell model. The protocell model is being used to investigate - in a bottom-up way - the possible roots of cellular autonomy, and the lipid compartment sub-system plays an integral part in determining the cellular dynamic. Whilst motivation is to ground the early model simulation results, the analysis here also reveals an interesting finding: simple addition of an 'osmotic buffer' to the lipid compartment not only widens it's range of stability, but also causes a profound change in the deep dynamical structure of the whole cellular system. Relevant to the origins of life, this bifurcation increases the robustness of the compartment to perturbation and instantly grants a richer behavioural repertoire including more reliable divide cycles.

1 Introduction

In recent work, Ruiz-Mirazo and Mavelli have presented a stochastic protocell model involved in a wider research programme investigating the minimal origins and graded appearance of cellular autonomy [6,7]. The membrane of the model is crucial in determining the system-wide dynamic, and in it's simplest form, as a bare lipid compartment, can be analysed quite readily. This work takes up this challenge and reveals 3-dimensional phase portraits for non-buffered and buffered lipid compartments. These dynamical pictures can be used to consolidate the 'grass roots' of the protocell model and explain many of the early results [4]. Additionally, insight gained here is hoped to carry across and leverage future analysis of the more complex lipid-peptide elaborations of the protocell. A more elaborate version of this work can be found in [1].

2 Compartment System for Analysis

The lipid compartment of the protocell model is based on an abstract notion of a lipid bi-layer (Fig. 1). The bi-layer is assumed to self-organise itself and then exist as a dissipative structure (within limits), continuously exchanging lipid molecules L with the core and environment through four flux events.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 230–239, 2011.

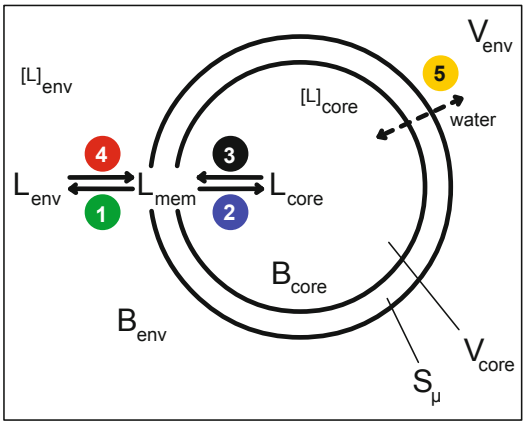


Fig. 1. Lipid compartment system for analysis

Usefully, this scenario enables the compartment to be accurately identified as a dynamical system of three variables: L_{core}, L_{mem} and L_{env}, the number of lipid molecules in the core, membrane and environment respectively.

Dynamics of the protocell model run on a modified version [5] of Gillespie's Stochastic Simulation Algorithm (SSA), [2,3]. The protocell system is defined as a series of events, and the SSA actions these events at rates defined by their *propensity* or 'natural tendency' values[1]. The dynamical characteristics of the compartment system are accordingly determined by how lipid movement around the four flux events changes the propensity (rate) of those flux events.

Before beginning, a quick review of relevant formulae is necessary. Lipid is released from the membrane (events 1 and 2) at propensity linearly dependent on the number of lipid molecules in the membrane L_{mem}

$$a_{\text{lipid release}} = K_{\text{L}} L_{\text{mem}} \tag{1}$$

where release constant $K_{\text{L}} = 0.001$. Conversely, lipid is absorbed from the core and environment into the membrane (events 3 and 4) at propensity

$$a_{\text{lipid uptake}} = K_{\text{L}\mu}[L]_{\text{env/core}} S_{\mu} \tag{2}$$

where uptake constant $K_{\text{L}\mu} = 1.0$, $[L]_{\text{env/core}}$ represents the respective molar concentrations of lipid molecules in the environment and core, and S_{μ} is the membrane surface area, which in the case of a pure lipid bi-layer membrane is equal to

$$S_{\mu} = \frac{L_{\text{mem}} \alpha}{2} \tag{3}$$

where the head area of lipid molecules $\alpha = 0.5\text{nm}^2$.

[1] The only knowledge of the SSA required for this analysis.

The fifth event in the system[2] is an instantaneous flow of water through the membrane ensuring an isotonic relationship between core and environment. The volume of the core V_{core} is constantly resized[3] so that the total molar concentration of osmotic species L and B in the core is equal to the same total in the environment:

$$\frac{L_{\text{core}} + B_{\text{core}}}{N_A V_{\text{core}}} = \frac{L_{\text{env}} + B_{\text{env}}}{N_A V_{\text{env}}} \text{ at all times} \tag{4}$$

where B_{core} and B_{env} are the number of buffer B molecules in the core and environment respectively, $V_{\text{env}} = 5.23e^{-16}$ litres and N_A is Avogadro's constant. Buffer molecules cannot permeate the membrane (but do exert osmotic pressure across it).

Finally, viability of the compartment is ensured between two mechanical limits $0.9 \leq \phi \leq 1.1 \times \sqrt[3]{2}$ where stability parameter Φ is defined as the ratio between the membrane surface area and the core surface area, calculated from the (assumed spherical) core volume:

$$\Phi = S_\mu / \sqrt[3]{36\pi V_{\text{core}}^2} \tag{5}$$

Outside the lower and upper viability limits, the compartment bursts or divides respectively.

3 Lipid Space

Starting the analysis, **Lipid Space** is defined as the 3-space containing all combinations of L_{env}, L_{mem} and L_{core}. In order to become informative, firstly the **viability region** is drawn on this state space, showing the range of lipid states corresponding to a stable cell compartment. Secondly, the **lipid dynamics** are super-imposed: these arrows and equilibrium planes/lines show the likely phase or flow of the space at different points - where each state will *probably* evolve to in time[4].

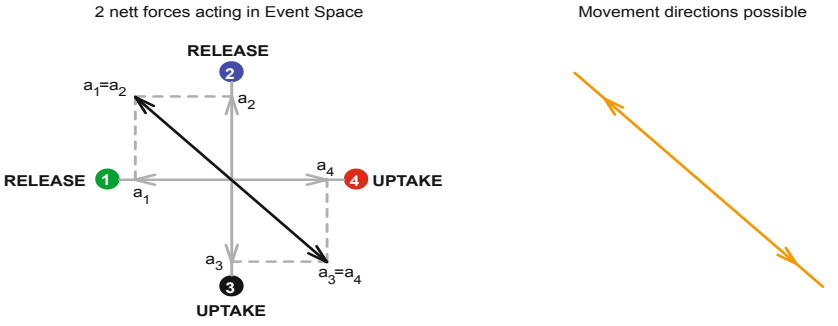
4 Viability Region

The viability region of the compartment is plotted onto the Lipid Space by substituting (3) and (4) into the stability criterion (5). The new equation defines a plane in Lipid Space

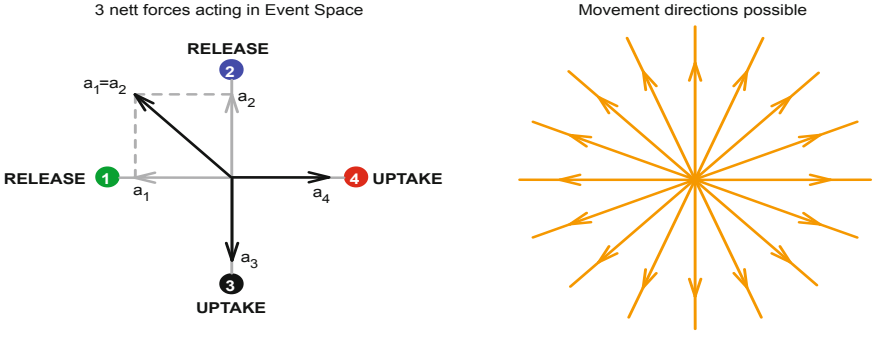
[2] Event 5 is not implemented as a Gillespie event controlled by propensity, but is rather a condition ensured to be always true.

[3] The core volume is determined independently of the membrane surface area enclosing it.

[4] Phase spaces in this analysis are **stochastic**. Rather than dictating the definite time evolution of the system (i.e. like phase arrows on a deterministic phase plot), phase arrows here just indicate the nett 'persuasion' acting on the system at a point. The absence of arrows does not mean there is no pull, rather it means that pull in all directions is equally likely.



(a) Event Space for non-buffered compartment.



(b) Event Space for buffered compartment.

Fig. 2. Nett reigning forces acting in Event Space for (a) non-buffered and (b) buffered compartments. Buffered compartments have a richer dynamical structure because three forces steer movement in Event Space, rather than just two opposing forces.

$$L_{\text{mem}} = \frac{2\Phi}{\alpha} \sqrt[3]{36\pi \left[\frac{L_{\text{core}} + B_{\text{core}}}{L_{\text{env}} + B_{\text{env}}} V_{\text{env}} \right]^2} \tag{6}$$

Setting the stability factor to the burst limit $\Phi = 0.9$ and plotting L_{mem} for all values of L_{core} and L_{env} gives the burst plane. Setting the stability factor to the divide limit $\Phi = 1.1 \times \sqrt[3]{2}$ and doing the same gives the divide plane. The 'sandwich' region in the Lipid Space between the two planes is the set of lipid states yielding a stable cell[5].

[5] Non-buffered cells, regardless of size, share the same two viability planes. However, buffered cells of different radii require different numbers of buffer molecules in the core to achieve a specific initial $[B]_{\text{core}}$ concentration, and so have different shaped viability planes.

5 Equilibrium States

The first step in developing the lipid dynamics is to identify the equilibrium states in the Lipid Space. A cell compartment is said to be in 'lipid equilibrium' when the membrane is absorbing lipid from the core and environment at the same rate it is releasing it to them. Equating (2) with (1), then substituting surface area (3) gives

$$[L]_{\text{core/env}} = \frac{2K_{\text{L}}}{K_{\text{L}\mu}\alpha} = \frac{2(0.001)}{1(0.5)} = 0.004M \tag{7}$$

So, as Mavelli and Ruiz-Mirazo state ([4], p1793), the concentration of lipid in the core and environment has to be 0.004M to achieve equilibrium. Now going further, when equilibrium states are plotted for non-buffered cells on the Lipid Space, they fall on a *plane*. A plane is defined because whereas the fixed volume environment requires exactly

$$L_{\text{env}} = 0.004 N_A V_{\text{env}} = 1259832 \tag{8}$$

lipid molecules to achieve $[L]_{\text{env}}$=0.004M, the core can theoretically contain any number of lipids: in the absence of buffer, osmotic balance (4) resizes the core volume ensuring $[L]_{\text{core}} = [L]_{\text{env}}$ at all times[6].

For buffered compartments, the equilibrium plane transforms into a *line* in the Lipid Space. The presence of buffer molecules (internal, external or both) makes the osmotic criterion (4) more general, and there becomes scope for $[L]_{\text{core}} \neq [L]_{\text{env}}$. To achieve $[L]_{\text{core}}$=0.004M then, L_{core} has to additionally assume a specific value too

$$L_{\text{core}} = \frac{KB_{\text{core}}}{1-K} \tag{9}$$

where

$$K = \frac{0.004 N_A V_{\text{env}}}{L_{\text{env}} + B_{\text{env}}}$$

Of note, in both non-buffered and buffered cases, lipid equilibrium is independent of the number of lipids in the membrane L_{mem}.

6 Event Space

In addition to moving in Lipid Space, the compartment system can also be thought to move in a 2-dimensional **Event Space**. Whereas Lipid Space represents the exact lipid state of the system, Event Space represents which of the four flux events have occurred from an initial lipid state[7].

[6] Of course, the L_{core} value has to remain within the cell viability region though.

[7] Event Space utilises the positive and negative X and Y axes to represent the four different events. The events do not require their own dimensions because event 4 (positive X) is simply the reverse of event 1 (negative X), and event 2 (positive Y) the reverse of event 3 (negative Y).

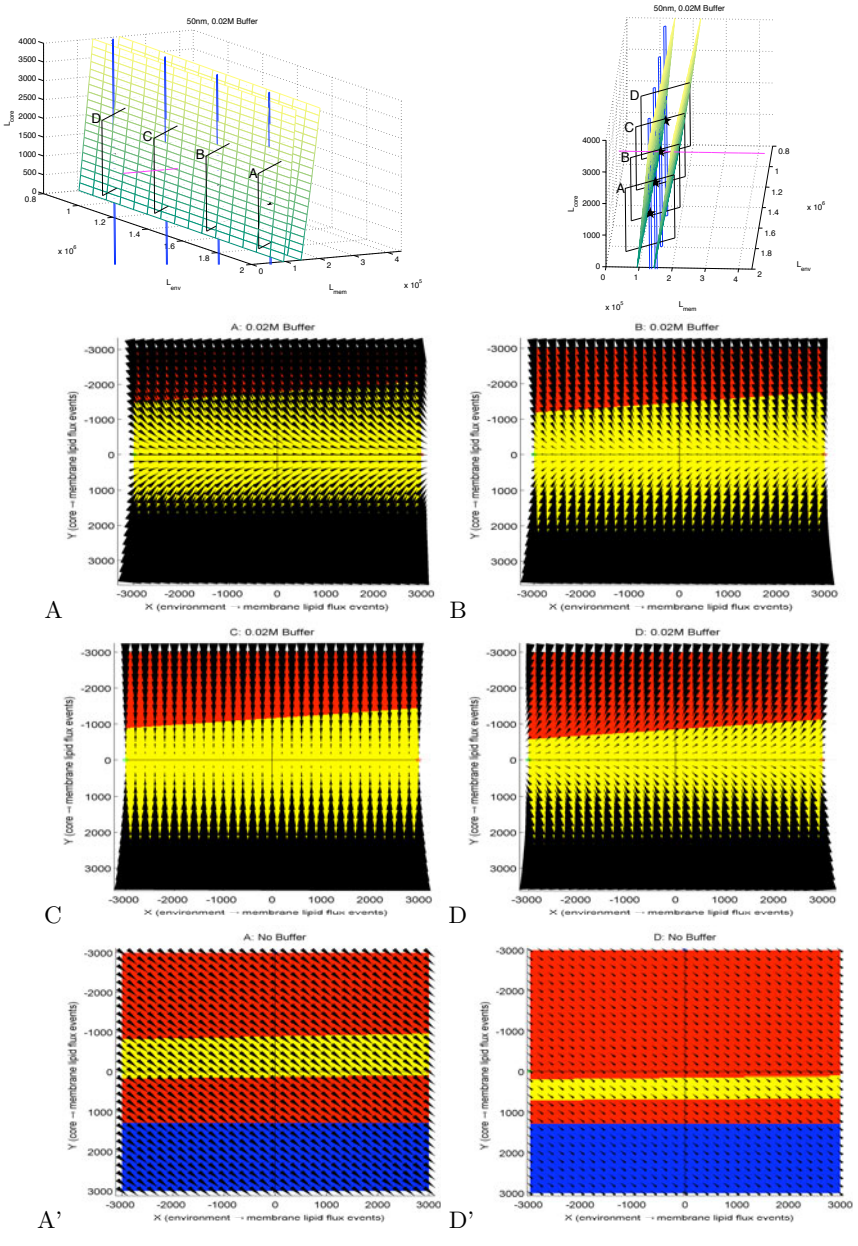
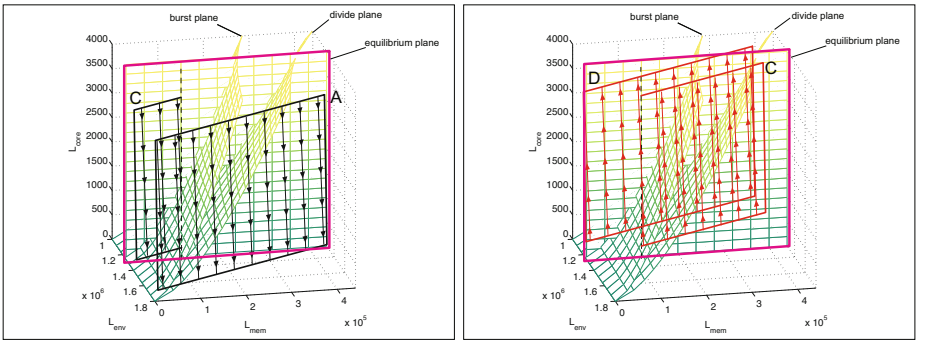
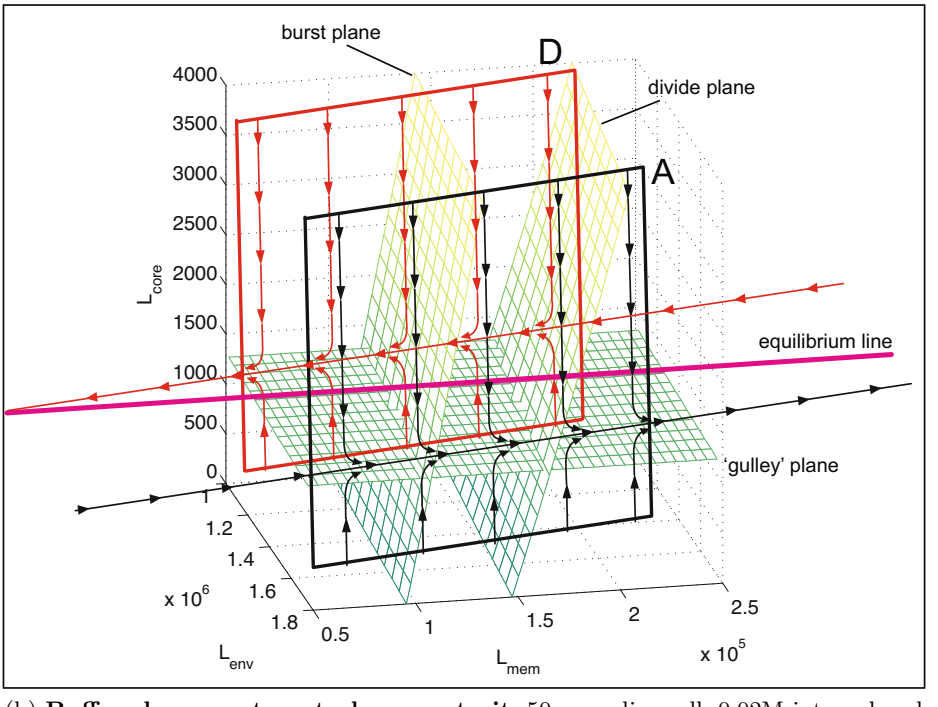


Fig. 3. Event Space slices with phase arrows super-imposed taken at positions A-D in the Lipid Space (top) of a 50nm 0.02M buffered compartment. For comparison, A' and D' show slices A and D for a non-buffered compartment.



(a) **Non-buffered compartment phase portraits**. 50nm radius cell, 0M internal and external buffer. Red lines in the right plot actually lie *behind* the equilibrium plane and would be obscured by it.



(b) **Buffered compartment phase portrait**. 50nm radius cell, 0.02M internal and external buffer.

Fig. 4. Illustration of (a) non-buffered and (b) buffered compartment phase portrait. Addition of buffer causes a spectacular (and favourable) bifurcation in the deep dynamical structure of the lipid compartment system: a 'gully' plane emerges as a strong basin of attraction to the equilibrium states

Event Space is an intuitive technique for drawing phase arrows onto 3-dimensional Lipid Space. As each flux event changes the compartment state in a pre-defined way, then Event Space and Lipid Space have a straightforward geometric mapping. Additionally, plotted in Lipid Space, an Event Space plane shows which range of lipid states (from an initial state) a compartment can naturally enter into through it's own lipid flux (when no external processes change lipid numbers in the system).

Plates A, B, C and D of Figure 3 are event space 'slices' taken through the Lipid Space (top of figure) of a 50nm radius compartment with 0.02M internal and external buffer[8]. Phase arrows have been drawn on each event space grid by calculating the propensities for the four flux events at each event point (x,y) and then drawing an arrow pointing in the mean vector direction of the next likely event. The size of arrow heads depict relative dynamic force. As comparison, the phase at points A and D for a 50nm non-buffered compartment are shown on plates A' and D' respectively.

Consolidating these results[9], Figure 4 illustrates the general 3-dimensional phase portraits for buffered and non-buffered compartments. Interestingly, the dynamical structure in each case is *profoundly* different.

7 Compartment Dynamics

7.1 Non-buffered Dynamics

On the one hand, non-buffered compartment dynamics are fairly unremarkable. The single molecular species - lipid - controlling the osmotic balance across the membrane proves to be a limitation.

With no buffer, moving lipid around the system through the four flux events only ever results in *two nett reigning forces* in Event Space (Fig. 2a). Propensities for uptake events 3 and 4 always match ($a_3 = a_4$) because osmotic equalisation (4) ensures $[L]_{\text{core}} = [L]_{\text{env}}$ at all times. Likewise propensities for release events 1 and 2 always match ($a_1 = a_2$) by definition of the model. Therefore, dynamic forces can push the system in just two movement directions in Event Space.

The limited dynamics mean that, when a sudden lipid increase or decrease is added to the environment from equilibrium conditions, a non-buffered compartment will flow straight to a divide or burst condition respectively. Secondly of note, the equilibrium condition is *weakly held*. In the local vicinity of the equilibrium plane, the nett dynamic forces acting on the system (for a 50nm radius compartment) are minute. Therefore 'at equilibrium' a non-buffered compartment is actually prone to wander and is quite unstable.

7.2 Buffered Dynamics: Bifurcation!

Introducing an osmotic buffer changes the dynamical structure of the compartment system in a spectacular way. The phase arrows in the Lipid Space become

[8] A common configuration used in the Ruiz-Mirazo and Mavelli protocell work.
[9] And other results/phase plots not shown.

totally re-organised and now flow to a 'gully' which acts as a basin of attraction to the equilibrium states that, as mentioned before, now fall on a line (Fig. 4b). The bifurcation is caused by the buffer molecules participating in the osmotic balance (4) enabling $[L]_{\text{core}}$ to become de-coupled from $[L]_{\text{env}}$. The knock-on effect from a dynamics perspective is that propensities for uptake events 3 and 4 need not necessarily be equal anymore, and *three nett reigning forces* operate in Event Space (Fig. 2b). Combinations of these three forces can produce a nett dynamic force pushing the system in any possible direction in Event Space, and this simple fact endows the compartment with potential for richer overall dynamics.

The 'gully' flows directly left and right on Event Space plots A-D (Fig. 3) and therefore occurs when the propensity for lipid release events 1 and 2 is equal to the propensity for lipid uptake event 3. When drawn on Lipid Space, the latter condition describes a plane with equation

$$L_{\text{env}} = \frac{2K_{\text{L}}N_A V_{\text{env}}(L_{\text{core}} + B_{\text{core}})}{K_{\text{L}\mu}L_{\text{core}}\alpha} - B_{\text{env}} \tag{10}$$

and passes through the line of equilibrium states.

With respect to non-buffered systems, the 'gully' feature of buffered compartments grants:

1. **Increased robustness to external lipid peturbations.** The gully creates a wide basin of attraction, providing a reliable route back to the equilibrium states from a wide variety of other lipid states. Equilibrium is *strongly held*.
2. **A more reliable divide cycle.** The gully provides a slower, more controlled route to compartment divisions or bursts, and thus a stable divide cycle can be established ([4],p1797,Fig. 5). Conversely, a non-buffered compartment cannot sustain a divide cycle.
3. **A richer dynamical substrate.** Buffered dynamics create a platform which other cellular processes can leverage to create further complex behaviour.

8 Brief Conclusions

This analysis forged an analytical/geometric way to think about the dynamic behaviour of the (bare) lipid compartment belonging to the Ruiz-Mirazo and Mavelli protocell model. A huge difference was found in the dynamic portraits of non-buffered and buffered compartments. The latter were found both more robust and enabled for richer dynamic behaviour, and these properties stemmed from buffer species being able to regulate water flow across the membrane in addition to lipid. In this way, addition of buffer could suggest an extremely rudimentary yet extremely effective improvement a bare lipid compartment could make at the very beginning of the road to increased levels of autonomy.

References

1. Blundell, B.J.: Forging the Road Toward Autonomy: Explorations with a Stochastic Protocell Model. Unpublished MSc thesis. University of Sussex, Brighton (2009)
2. Gillespie, D.T.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of Computational Physics 22(4), 403–434 (1976)
3. Gillespie, D.T.: Stochastic Simulation of Chemical Kinetics. Annu. Rev. Phys. Chem. 58, 35–55 (2007)
4. Mavelli, F., Ruiz-Mirazo, K.: Stochastic simulations of minimal self-reproducing cellular systems. Philosophical Transactions of the Royal Society B 362, 1789–1802 (2007)
5. Mavelli, F., Lerario, M., Ruiz-Mirazo, K.: ENVIRONMENT: a computational platform to stochastically simulate reacting and self-reproducing lipid compartments. Physical biology 7(3), 036002 (2010), `http://www.ncbi.nlm.nih.gov/pubmed/20702920`, doi: 10.1088/1478-3975/7/3/036002, ISSN: 1478-3975
6. Ruiz-Mirazo, K., Mavelli, F.: Simulation model for functionalized vesicles: Lipid-peptide integration in minimal protocells. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 32–41. Springer, Heidelberg (2007)
7. Ruiz-Mirazo, K., Mavelli, F.: On the way towards 'basic autonomous agents': Stochastic simulations of minimal lipid-peptide cells. BioSystems 91, 374–387 (2008)

The Role of the Spatial Boundary in Autopoiesis

Nathaniel Virgo, Matthew D. Egbert, and Tom Froese

Centre for Computational Neuroscience and Robotics
University of Sussex, Brighton, UK
{nathanielvirgo,t.froese}@gmail.com, mde@matthewegbert.com

Abstract. We argue that the significance of the spatial boundary in autopoiesis has been overstated. It has the important task of distinguishing a living system as a unity in space but should not be seen as playing the additional role of delimiting the processes that make up the autopoietic system. We demonstrate the relevance of this to a current debate about the compatibility of the extended mind hypothesis with the enactive approach and show that a radically extended interpretation of autopoiesis was intended in one of the original works on the subject. Additionally we argue that the definitions of basic terms in the autopoietic literature can and should be made more precise, and we make some progress towards such a goal.

1 Introduction

The idea of autopoiesis is a venerable part of the artificial life tradition. Ideas from the theory of autopoiesis formed part of the foundations on which the field of artificial life was built [2] and have been widely cited ever since.

However, over its lifetime the idea of autopoiesis has meant different and in many cases quite incompatible things to different authors. An important part of the subject's maturation will be to determine more precisely whether these alternative interpretations are compatible with each other and what, if anything, forms their common theoretical core.

We suggest that much of the conflict in this field comes from the conflation of two concepts that should be kept distinct: the *physical boundary* of an autopoietic system, which is produced by the system and makes an important contribution to the working of the system; and what we call its *operational limits*, which determine which processes are part of the system. The goal of this paper is to make these concepts, and the distinction between them, as clear as possible. Failure to do this in the past has lead to a kind of internalism in which the network of processes that constitute an organism is seen to lie entirely within its physical boundary, an idea that sits uncomfortably with the conception of cognition as relational. We believe that by clarifying this distinction and introducing new terminology for it we will make a direct contribution towards current modelling and theoretical work.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 240–247, 2011.

Enaction and the Extended Mind. One particular point of relevance for this discussion is a current debate about whether the extended mind hypothesis [3] is compatible with the 'enactive' approach developed by Varela and colleagues [9], in which both autopoiesis and an extended approach (in which cognitive processes can take place outside an organism's physical bounds) play central roles. The possibility of incompatibility between the two was first raised by Wheeler in a talk at last year's Artificial Life conference [10,11], which has subsequently been the target of a critical analysis by Di Paolo [5].

Wheeler's argument forms a useful point of departure for clarifying the way in which enactive cognitive science conceives of the complex relationship between life and mind, as well as its operational understanding of cognition. It progresses with the following steps:

1. Autopoiesis is a non-negotiable component of enactive cognitive science.
2. Autopoiesis is a type of self-organization defined by the production of a physical boundary that distinguishes the system as a material unity.
3. One interpretation of the primary literature is that "autopoiesis = life = cognition."
4. It seems to follow from steps 2 and 3 that the enactive approach is committed to the claim that the cognitive system is co-extensive with the living system, entailing that both are bounded by the living system's physical membrane.
5. Since cognition is therefore internal to the physical boundary of the autopoietic system, the enactivist cannot endorse the extended mind hypothesis.

Since the autopoietic and enactive traditions have always insisted on the relational nature of cognition as emerging out of the dynamics of a brain-body-world systemic whole, this conclusion might come as a surprise. Is the enactive approach really committed to the claim that cognition is something that happens within the spatial bounds of an organism?

One way to dissolve this particular incompatibility is to admit that the "life = cognition" slogan has outlived its usefulness. This is the approach pursued by Di Paolo [5], who emphasises the non-reducibility (non-intersection) but mutual interdependence of the metabolic (constitutive) and cognitive (relational) domains of discourse in the autopoietic tradition. On this view the enactive approach to cognition is committed to neither an internalist nor an externalist position: "as relational in this strict sense, *cognition has no location*." [5, p. 19, original emphasis].

Though compatible with Di Paolo's argument, our position has a different focus. It is Wheeler's interpretation of autopoiesis as a self-sustaining network of molecular processes that occur *within* a physical boundary (step 2 above) that gives him the original motivation for his argument. This internalist interpretation of autopoiesis may indeed be held by some of the idea's current proponents, but we argue for a different interpretation, in which the physical 'constitutive' processes by which an organism's structure is produced need not all take place within its physical boundary. By spelling out in detail the distinction between the spatial boundary and the operational limits we arrive at a view of enactive

cognitive science that cannot be considered 'internalist' neither on the cognitive nor the physical, metabolic level.

The Changing Definition of Autopoiesis. It is important to be clear that autopoiesis is not a well-defined concept, even though it sometimes appears to be. Much of the primary and secondary literature is written in a style that suggests the theory being discussed is fully developed and quite formally defined; but in fact the meaning of many of the key terms, including the word 'autopoiesis' itself, change quite fluidly from publication to publication (see [1]). For instance, in [7] autopoiesis is explicitly presented as a theory that applies to all life, whether multicellular or unicellular, whereas in [8] and later publications the idea is said to apply only to single cells, with multicellular organisms requiring a special 'second-order' autopoiesis.

In order for the theory to be moved forward it is important to continue to work on these definitions. It is not enough to quote one of the varying definitions that Maturana and Varela gave, since these were neither precise nor unchanging. Moreover they depend on the meanings of words such as 'process' which as far as we are aware were never defined in any of the primary literature [6]. Our approach is to try to reveal some of the key concepts by stripping away some of the convoluted and overly formal language, focusing instead on sharpening our (pre-formal) intuitions.

2 Defining Operational Closure and Its Limits

What is a process? Since the word 'process' has such a key role in all of Maturana and Varela's definitions of autopoiesis it is surprising that little seems to be written about its precise meaning (though see [4]). A related under-asked question is what it means for a process to be *enabled by* or *dependent upon* another process. Since our discussion below also hinges heavily on the concept, we will briefly summarise our (somewhat tentative) intuitions about what a definition might look like here.

A tentative definition of a process might be that it is something that happens repeatedly, or which tends to happen whenever the right conditions are met (examples of processes that meet this definition include: the fermentation of sugar into alcohol; diffusion of heat from hot to cold bodies). There are several properties that are shared by every process, at least in the physical/chemical domain: every such process transforms something into something else (a chemical reaction transforms its reactants into its products; a transport process transforms the spatial distribution of a substance; friction transforms kinetic energy into heat). All such processes also have conditions which must be met in order for them to take place, or which affect the rate at which they occur. These can be trivial (e.g. simply the presence of the reactants) or more complex (such as the presence of enzymes and a specific temperature).

Note that on this definition, processes are separate from the dynamics: the dynamics are the ways in which variables change over time, whereas processes

are things that cause them to change. Thus one can model the dynamics of a system without modelling the processes that underly them.

Importantly, the operation of a process can modify the conditions that determine whether another process takes place. Rather than a process B depending directly upon another process A, we have a situation in which process A produces something which is a required condition for process B to occur. Process B then depends on A, via the conditions that process A helps to generate. These relationships of dependence can form networks of processes with the interesting property of operational closure which we shall now discuss.

Operational Limits and Spatial Boundaries. In this section we introduce the term *operational limits* to describe which processes should be seen as an operational part of a system. We discuss the relation between operational limits and *spatial boundaries* and show why the two notions are often conflated, despite being quite distinct.

To define the operational limits of a system, it is necessary to understand the notion of *operational closure*. Figure 1 depicts a hypothetical system of processes that are dependent upon or enabled by each other. These interdependencies are depicted in the figure as arrows connecting the processes. Given such a network of processes and relationships of dependence we can define which parts of the network are operationally closed. However, before we give this definition, we think that it is important to point out that its application requires, as a precondition, the identification of all of the processes and relationships of dependence. As we mentioned in the previous section, at this stage, neither 'process' nor 'dependence' (sometimes referred to as 'conditioning' or 'enabling') are well defined. Thus far, researchers have depended upon intuitive understanding of these phenomena, but they are in need of formalization if we are to consider operational closure rigorously defined.

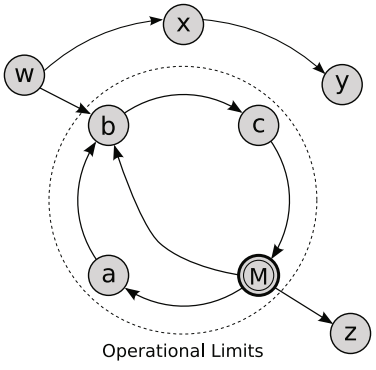


Fig. 1. A hypothetical network of processes connected by interdependencies. Lettered circles represent processes and arrows represent 'enable'. M represents a process that generates the spatial boundary.

Definition. Given a collection of processes $\mathcal{C}$, we can identify an operationally closed subset of those processes, $\mathcal{S}$ such that for every constituent process P, the following conditions are true.

1. Another process P' requires conditions produced by process P.
2. Process P is conditioned by another process P''.
3. P' and $P'' \in \mathcal{S}$.
4. P' and P'' can be (but are not required to be) the same process.

In graph theory terms, this defines a strongly connected subgraph of the directed graph of process dependencies. Assuming that all of the processes and interdependencies are included in Figure 1, processes w, x, y and z are not part of any operationally closed network. This is the case because each one of these processes does not depend upon another process which is in turn dependent upon the original process. Take as a case in point, process x which is dependent upon w which is not dependent upon any process in the system. An absence of cyclical dependence indicates an absence of operational closure. In contrast, processes a, b, c and M form an operationally closed network. Process c depends upon b which depends upon a which depends upon M which depends upon c, closing the loop and making the set of four processes operationally closed. A second, smaller operationally closed loop exists, consisting only of processes M, b, and c. These are the only operationally closed loops within this system.

It is not difficult to find examples of processes that have cycles of dependency on a variety of scales from subsystems of an organism to relationships of dependence on an ecological or global scale. Furthermore, operationally closed networks of processes may or may not involve organisms at all. This is not problematic for the theory of autopoiesis as autopoietic systems are the subset of operationally closed systems that produce a spatially bounded structure.

Let us now consider this spatial boundary. There is no requirement that any of the above processes occur inside or outside of the spatial boundary. It may be tempting to think that because processes a, b, c and M are all inside the operational limits, they are also inside the spatial boundary, but this is not necessarily the case. It might be that only processes a, b, and z are inside the spatial boundary. The spatial boundary is not the same as the operational limits.

In a way cell membranes, or spatial boundaries in general, seem similar to the operational limits in that they define a boundary inside which certain processes lie and others do not. For this reason, it has been tempting for some authors to depict spatial boundaries on relational diagrams as a circle surrounding a number of processes, similar to the depiction of the operational limits of an organization. To do this however is to commit the error of conflating operational limits with spatial boundaries. Figure 1 is not drawn in physical space – it is a relational diagram of processes. As such, it is inappropriate to depict boundaries in their spatial form (e.g. as an encircling). It is only appropriate to depict them as yet another process (e.g. process M in our diagram) or set of processes that has various interdependencies with the other processes in the network.

On one hand this error does not seem too serious if, for example, one is drawing informal diagrams intended to get a point across. However, there are a number of serious conceptual errors that can be caused by confusing the spatial and relational domain. For example, processes can span the spatial boundaries of an organism (e.g. ion pumps in cell-membranes or the production of heat by warm-blooded animals affecting the animal's environment). This possibility is lost when spatial boundaries and process relationships are conflated and physical and relational structures are plotted in the same space.

A related error is the inappropriate inflation of the importance of the spatial boundary. The spatial boundary is undoubtedly important in maintaining the conditions necessary for many ongoing processes in living organisms. While these are indeed important contributions, we do not believe that they are of a different *type* of contribution than the other enabling processes that form living organisms. In fact, we believe that the inflation of the importance of the spatial boundary runs contrary to one of the more provocative ideas to come from the autopoietic school of thought. Namely, that the spatial boundary of the organism is not actually equivalent to the limits of the organism – that the organism, as an autopoietic system, includes processes that are not occuring within its spatial boundary.

3 Extended Autopoiesis

In this section we defend our claim that the operationally closed network that constitutes an autopoietic unity can include processes that occur outside of its spatial boundary by showing that this was the interpretation intended in one of the earliest pieces of literature on the subject, Maturana and Varela's *Autopoiesis and Cognition* [7].

This does not validate our claim entirely; what matters to science is what is useful to us in the present day, not the precise words that were first written 37 years ago. However the original exposition is quite clear and we hope that by re-examining it with a simple example we can better express our own perspective.

In [7], Maturana and Varela introduce us to the concept of homeostatic machines. These are defined as machines which maintain constant, or within a limited range of values, some of their variables, a definition which will be familiar to most of us. However this definition is followed by an important clarification which we take as fundamental to how the rest of the theory is to be interpreted. Since the clarification of this definition is so important we quote the whole paragraph:

> "There are machines which maintain constant, or within a limited range of values, some of their variables. The way this is expressed in the organization of these machines must be such as to define the process as occurring completely within the boundaries of the machine which the very same organization specifies. Such machines are homeostatic machines and all feedback is internal to them. If one says that there is a machine M, in which there is a feedback loop through the environment so that the effects of its output affect its input, one is in fact talking about a larger machine M' which includes the environment and the feedback loop in its defining organization." [7, section 1.2.a]

This can be clarified with an example. Let us consider a mechanical thermostat. This is an archetypal example of a homeostatic machine (though of course it is not autopoietic). The variable which it keeps within bounds is the temperature of a room. However, according to the quoted paragraph it is not correct (in the autopoietic language) to think of the thermostat as being the box on the wall

that is connected to a heater and contains a thermocouple, because this machine (machine M) has a feedback loop that runs through the environment. When the temperature drops, the thermocouple breaks a connection, which causes the heater (not part of machine M) to be switched off, causing the temperature to drop again. Since the thermostat relies on this feedback loop for its operation, we should actually define the thermostat as a larger machine (machine M') which includes the heater, the air in the room, and the feedback loop that passes through them.

Why is this so important? The above quoted paragraph is positioned directly before the definition of an autopoietic machine is spelled out[1], and just below that we are given the following key statement:

> "Therefore, an autopoietic machine is an homeostatic (or rather relation-static) system which has its own organization (defining network of relations) as the fundamental variable which it maintains constant." [*ibid.*]

Autopoietic systems, then, are to be seen as homeostatic machines. It follows that their definition must be expanded in the same way if they rely on a feedback loop that runs through their environment.

Wheeler [10] gave the example of an earthworm, which builds tunnels held open by a sticky secretion that helps to digest its food. We can try to see the worm as an autopoietic system (and hence an homeostatic system) whose operational limits are defined by its physical boundary (its skin). However, the worm relies on the effects of its secretions; this is a feedback loop which runs through its environment. The above quoted paragraph from [7] thus compels us to redefine the system so that it includes not only the worm itself but also the secretions and their effects. On this view the autopoietic system that constitutes the worm is not coextensive with the unity that we refer to as "the worm," it is much bigger. This will be the case for most if not all organisms, since most organisms rely not only on sensory-motor loops that run through their environment but also on nutrients that are recycled externally to them.

4 Future Considerations

In the past, because it is easily visualizable, because terminology was confusing, and perhaps because of difficulties associated with translation, we have seen the

[1] The full definition of autopoiesis as given in [7] is as follows. Note that this version of the definition hinges on the constitution of a concrete unity in space but does not specify that this unity must be bounded by a distinct membrane.

> An autopoietic machine is a machine organized (defined as a unity) as a network of processes of production (transformation and destruction) of components that produces the components which: (i) through their interactions and transformations continuously regenerate and realize the network of processes (relations) that produced them; and (ii) constitute it (the machine) as a concrete unity in the space in which they (the components) exist by specifying the topological domain of its realization as such a network.

physical boundary of an autopoietic system as playing a special role in both the physical and the relational domains. But here we have argued that in the relational domain the spatial boundary should take its place among the other enabling conditions. In the physical domain it plays an important role in helping to define the organism as a distinct unity but it plays no special role in the relational domain, except perhaps in that it enables a great number of processes.

We have shown some of the implications of this for the debate about the compatibility between autopoiesis and the extended mind hypothesis, and we believe that it is relevant to much current theoretical and modelling work.

We have also begun working towards precise definitions of some basic concepts in the autopoietic theory, which were previously absent. The definitions we have outlined are tentative. However we have tried to express them in such a way that the intended interpretation is clear. We hope that future authors will try to give similarly precise definitions of basic terms. The theory of autopoiesis can only become stronger as a result.

Acknowledgements

This paper arose from discussions at the Life and Mind seminars at Sussex, and on the associated blog (http://lifeandmind.wordpress.com). The contribution of Ezequiel Di Paolo to this discussion was particularly influential.

References

1. Bourgine, P., Stewart, J.: Autopoiesis and cognition. Artif. Life 10, 327–345 (2004)
2. Bourgine, P., Varela, F.J.: Introduction: Towards a practice of autonomous systems. In: Varela, F.J., Bourgine, P. (eds.) Proc. ECAL 1. MIT Press, Cambridge (1992)
3. Clark, A., Chalmers, D.: The extended mind. Analysis 58, 10–23 (1998)
4. Di Paolo, E.A.: Artificial life and historical processes. In: Kelemen, J., Sosík, P. (eds.) ECAL 2001. LNCS (LNAI), vol. 2159, pp. 649–658. Springer, Heidelberg (2001)
5. Di Paolo, E.A.: Extended life. Topoi 28(1), 9–12 (2009)
6. Froese, T., Virgo, N., Izquierdo, E.: Autonomy: A review and a reappraisal. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 455–464. Springer, Heidelberg (2007)
7. Maturana, H.R., Varela, F.J.: Autopoiesis and Cognition: The Realization of the Living. Kluwer Academic Publishers, Dordrecht (1980)
8. Maturana, H.R., Varela, F.J.: The Tree of Knowledge: The Biological Roots of Human Understanding. Shambhala Publications, Boston (1987)
9. Thompson, E.: Mind in Life: Biology, Phenomenology and the Sciences of Mind. The Belknap Press of Harvard University Press, Cambridge (2007)
10. Wheeler, M.: Autopoiesis, enactivism, and the extended mind (abstract). In: Bullock, S., et al. (eds.) Proc. ALIFE XI, p. 819. MIT Press, Cambridge (2008)
11. Wheeler, M.: Minds, things and materiality. In: Renfew, C., Malafouris, L. (eds.) The Cognitive Life of Things: Recasting the Boundaries of the Mind. McDonald Institute for Archaeological Research Publications, Cambridge (in press)

Chemo-ethology of an Adaptive Protocell

Sensorless Sensitivity to Implicit Viability Conditions

Matthew D. Egbert, Ezequiel A. Di Paolo, and Xabier E. Barandiaran

Evolutionary and Adaptive Systems Group,
CCNR, University of Sussex, Brighton, BN1 9QJ, UK
mde@matthewegbert.com, ezequiel@sussex.ac.uk,
xabier.academic@barandiaran.net

Abstract. The viability of a living system is a non-trivial concept, yet it is often highly simplified in models of adaptive behavior. What is lost in this abstraction? How do viability conditions appear in the first place? In order to address these questions we present a new model of an autopoietic or protocellular system simulated at the molecular level. We propose a measurement for the viability of the system and analyze the 'viability condition' that becomes evident when using this measurement. We observe how the system behaves in relation to this condition, generating instances of chemotaxis, behavioural preferences and simple (yet not trivial) examples of action selection. The model permits the formulation of a number of conclusions regarding the nature of viability conditions and adaptive behaviour modulated by metabolic processes.

1 Connecting Biological Organization and Adaptivity

Conceptualizing adaptivity (the capacity of a system to cope flexibly with its environment in order to survive and reproduce) is far from trivial. The widespread strategy is to model adaptivity as the optimization of certain parameters (captured by the notion of fitness) or as the maintenance of certain variables (often called essential variables—[1]) within viability limits. As a consequence, models of adaptive behavior generally fall under one of two categories (or a combination of both). *Externalist*: Optimization techniques are used to constrain the behavior of a system to achieve the desired adaptive coupling with its environment in relation to a set of parameters or "fitness" criteria. This category includes different types of supervised learning algorithms for NN, simulated annealing or artificial evolution techniques to design control architectures (as used in Evolutionary Robotics, [2]) or, simply, hand design. *Internalist*: Models belonging to this class incorporate a set of internal variables often interpreted as energy sensors, pain or pleasure indicators, etc. These "value modules" are then coupled to other control mechanisms in order to tune the behavior of the system (as in reinforcement learning) or to choose between competing possibilities for action (acting as an action selector [3,4]).

In both cases the parameters or functions to be optimized are explicitly represented either as an external fitness function or as an internal value module, abstractly measuring how well adapted/adapting the system is. There is

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 248–255, 2011.

generally no reference or feedback to the processes from which these criteria emerge. How those boundaries of viability or optimal values come to be there in the first place is rarely addressed and modelled. As Randall Beer recognizes, "this explicit separation between an animal's behavioral dynamics and its viability constraints is fundamentally somewhat artificial. (...) However (...) we can assume that its viability constraint is given a priori, and focus instead on the behavioral dynamics necessary to maintain that existence. [5, p.265]".

At first sight, and for many cases, this abstraction seems reasonable. For instance it is obvious that above a certain temperature value an organism will die or that without a certain quantity of resources it would cease to exist. However, these conditions (or value functions) are often variable and difficult to determine, they show temporal variability and subtle interactions with other processes (e.g. you can survive at a low temperature for some time but not for "too long" and this in turn might depend on your diet, etc.). Critically, the behavior of organisms might be sensitive to these conditions in many and sophisticated ways that are lost when a priori abstractions are made. For instance, organisms might display a complex dynamic interplay between internal and behavioral adaptive modulations where mechanisms of self-repair, growth, digestion and maintenance are integrated with behavior generating mechanisms in many subtle ways [6].

What happens when we remove this somewhat artificial and explicit "separation between an animal's behavioral dynamics and its viability constraints"? To address this question requires reference to more fundamental aspects of biological organization such as the the modelling of energy consumption processes, metabolic organization, generation of movement, etc. However, on this side of the relationship between behavioral adaptivity and living organization (dealing with the emergence of viability conditions) life is usually modelled without including behavioral adaptivity. These models emulate the biochemical processes that make viability conditions and value functions be there in the first place. They describe life as a networked set of chemical reactions (metabolism) continuously re-producing the conditions required for their existence. Standing in far-from-thermodynamic-equilibrium conditions and, therefore, in a continuous need for matter and energy for their maintenance, minimal protocells [7] (or autopoietic systems [8]) come to capture the fundamental root of adaptivity: the need to actively compensate for a decaying or precarious existence that also defines the fragile limits (viability conditions) of their otherwise dissipating organization. However, these types of models tend to place the system in environments which do not require any system-level regulation of interactions with the environment (behavior) to maintain themselves (e.g. [9]). A few recent models (see [10,11]) have begun incorporating mechanisms of system level behavior, e.g. motion, upon which the autopoietic processes depend. Yet, many aspects of the interplay between behavior and metabolism are still to be explored.

In this paper we present a model of minimal metabolism and motility in a protocellular system simulated at the molecular level. The model is rather minimal yet capable of raising conceptual issues around the nature of viability conditions, temporal aspects of adaptive processes and the mutual dependence

between metabolism and behavior. Section 2 presents details of the model. Section 3 presents a set of experiments with the protocell behaving adaptively by performing chemotaxis and showing emergent forms of action selection without explicit sensors. Finally we conclude with section 4 addressing some theoretical implications and future extensions of the present model.

2 A Chemo-ethological Model of an Adaptive Protocell

We take a chemo-ethological approach in our explorations: a combination of aspects of artificial chemistry and forms of behavioral modelling and analysis. Our model is a modified version of a model presented in [11] and can be thought of as a highly simplified model of a protocell[7]. It takes place in a two-dimensional arena 256 units square. The model is simulated at the molecular level. It comprises three types of interactants: metabolites, resources and a membrane that encapsulates the reaction network. The interactants are governed by a set of chemical reactions giving rise to a self-maintaining metabolic network. Interactions between the metabolites and the membrane endow the system with an ability to move around the environment which contains generators of the necessary resources.

Metabolites. A metabolite is specified by five attributes, x, y, s, d and T. x and y represent the metabolite's spatial position and s represents the size of the metabolite, which affects its rate of thermal motion (more below). The type of a metabolite, T, indicates which chemical reactions the metabolite can participate in. The final metabolite parameter, d, represents the stability of the metabolite. Each iteration, there is a chance ($p = 5d \times 10^{-3}$) that the metabolite disintegrates. As one would expect, metabolites of the same type have the same s and d values. The metabolites are simulated as if in Brownian motion using the following equations: $x_{t+\delta t} = x_t + \delta t(g_x + 0.75v_x)$, $y_{t+\delta t} = y_t + \delta t(g_y + 0.75v_y)$. Here v_x and v_y represent the 2D velocity of the membrane. g_x and g_y represent displacement due to thermal motion and are selected each iteration from a Gaussian distribution (mean 0, std. $0.1/s$) where s represents the size of the metabolite and indirectly its rate of thermal motion.

Reactions. Metabolites are governed by the reactions shown in Table 1. Each reaction has a rate (ρ) which determines the likelihood of the reaction occurring. Reactions are simulated by picking 2 metabolites within the simulation N times (where N is proportional to the number of metabolites in the simulation) and performing their reaction if they are within 2 units of distance from each other. Metabolites never exist outside of a cell membrane as they are created inside the cell and can not move through the membrane.

Resources. There are three types of resource (R_0, R_1, and R_2) that react with the metabolites (see Table 1). Resources are represented by a 64×64 lattice of squares of width 4.0 units, the nodes of which are updated according to the following differential equation which simulates diffusion. $d\phi(\boldsymbol{r}, t)/dt = D\nabla^2\phi(\boldsymbol{r}, t) + q(r)$. Where $\phi(\boldsymbol{r}, t) \in [0, 3]$ represents the concentration of the resource at location $\boldsymbol{r}$ at time t, and $q(r)$ represents the addition of resources

Table 1. Metabolite Types & Chemical Reactions. ρ_f and ρ_b represent the rate of chemical reactions in the forward and backward directions respectively.

Metabolite Types

Name	Size	Stability	ΔPhosph.	Δvel.
X	0.8	0.005	0.00	0.0
Y	0.8	0.005	0.00	0.0
Z	0.5	0.001	0.15	0.1

Reactions

#	$R_1+R_2 \leftrightarrow P_1+P_2$	ρ_f	ρ_b	κ
0:	$Z + R_0 \leftrightarrow Z + Z$	1×10^{-2}	0	0.7
1:	$X + R_1 \leftrightarrow X + Y$	1×10^{-2}	0	0.7
2:	$Y + R_2 \leftrightarrow Y + X$	1×10^{-2}	0	0.7
3:	$X + Y \leftrightarrow Z + Z$	5×10^{-3}	1×10^{-3}	n/a

to the environment at resource *generators* which are placed in different areas depending on the experimental scenario. The local concentration of resource is increased by a fixed amount every iteration. Resources can act as one of the reactants in a chemical reaction. The parameter κ indicates the quantity of resource consumed by the reaction.

Membrane and Motion. The membrane is specified by three attributes: x, y, and p. It is circular and centered at x and y with the parameter p representing the number of phospholipids in the membrane which is directly proportional to the circumference, relating the radius of the membrane to the number of phospholipids thus: $r = 20p/2\pi$. The number of phospholipids in a membrane decays exponentially according to the equation $dp/dt = -5p \times 10^{-4}$.

Upon contact with the membrane, metabolite Z both imparts an outward radial velocity to the membrane and becomes part of the membrane as a quantity of phospholipids. The other metabolites simply bounces off the membrane, being returned to a position slightly closer to the center of the cell. Each iteration the membrane's location is updated according to its velocity which is reduced each iteration by a fixed drag constant.

3 Exploring the Dynamics of the Protocell

Measuring Viability Conditions. The first, simplest scenario that we examine with our model is one in which the environment contains a fixed quantity of homogeneously distributed R_0. Inside this environment there is one protocell containing a number of Z metabolites. Z is auto-catalytic in the presence of R_0 (see Table 1). Also note that Z contributes phospholipids to the membrane and that this contribution is the only process that counteracts the continual degradation of the membrane. It follows that if R_0 is sufficiently high, the autocatalysis of Z will be sufficient to completely compensate for the degradation of the membrane. If not, the membrane will shrink until the cell dies[1]. Therefore, a good candidate for a viability condition across different environmental situations is

[1] The relationship between resource availability and membrane size is not as simple as it might first appear. A smaller membrane requires less Z-production to maintain its size, but also has non-linear effects upon the levels of resource that are available to the protocell.

the rate of production of Z in relation to the rate at which the membrane degrades, $\Delta V \equiv d(Z/\mathrm{p})/dt$ (where Z is the number of Z metabolites and p is the number of phospholipids in the membrane). Furthermore, $\Delta V = 0$ is an interesting reference as protocells that maintain a negative ΔV for an extended period of time will die, unlike those that maintain a ΔV of 0 or greater.

This can be seen in Figure 1 which depicts values of ΔV for agents in the fixed resource environment[2]. Thinner trajectories plotted in grey tended to die. Note the 'viability boundary' located at $\Delta V = 0$, dividing those trajectories that tend to live from those that tend to die. The viability measure, ΔV can be thought of as a measure of what would happen should the protocell remain in its current situation for a long time. Negative values indicated a propensity towards death and positive values indicate the opposite. Note that this viability condition of the system is not explicitly encoded (unlike classical approaches) but is rather a statistical measure of spatially distributed molecular processes.

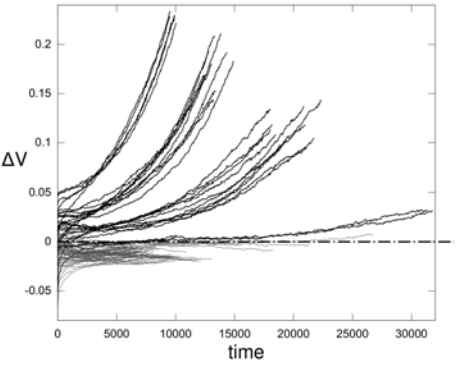


Fig. 1. System Viability

Experiment 1: Chemotaxis and its effect on viability. For the first experiment we move into a more complex scenario where rather than having a fixed homogeneous concentration of R_0, we utilize a R_0 generator which rotates through four different locations, moving every 5000 iterations. Figure 2 (left) shows the behavior of the protocell, which performs chemotaxis towards the generator. This motion is the result of the asymmetrical distribution of R_0 within the protocell. The portion of the protocell that has a higher concentration of R_0 will produce more Z. Accordingly, more Z particles will collide with the membrane in this area of the protocell, inducing an overall up-gradient motion.

Figure 2 (right) shows how ΔV oscillates above and below ($\Delta V = 0$). This plot indicates how the system is behaving adaptively; not in relation to an a priori and somewhat artificial parameter, but in relation to the very conditions upon which the system's ongoing survival depends. When the generator disappears, the ΔV becomes increasingly negative. This tendency is inverted by the system as it approaches the next generator. The protocell compensates for the negative tendency of ΔV by *behaving* (i.e. changing the conditions such that the ΔV becomes positive again).

Experiment 2: Oscillatory behavior between two generators. In our second experiment, we designed an environment in which even if resources are

[2] To generate this plot, the simulation was initialized with protocells with different starting conditions ($\#Z = \{50, 100, 150\}$, p= $\{8, 10, 12\}$, $R_0 = \{0.3, 0.4..0.8\}$) and we plotted the mean trajectory of 25 runs in ΔV over time (data was also smoothed using a 250 iteration running-mean low-pass-filter).

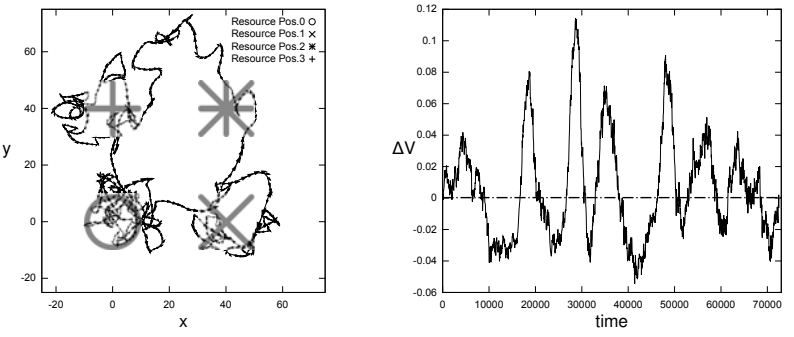


Fig. 2. Experiment 1, the protocell's response to a moving resource generator

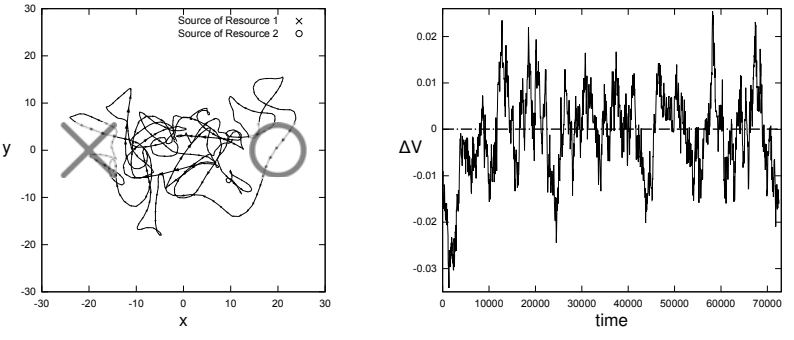


Fig. 3. Experiment 2, Dependence upon two different resources

fixed, the protocell can not survive without behaving – thereby forcing the system into a continuous transient of viability. We accomplished this by introducing two stationary resource generators; one of R_1 and one of R_2. The protocell oscillates back and forth between both generators (see Figure 3 left). Again the motion towards the relevant resource-source is produced primarily by the asymmetry within the cell of the production of Z. In this scenario, Z is only produced by an interaction between X and Y[3]. It is accordingly produced more in areas of the cell that are high in both X and Y than areas that have low concentrations of one of these metabolites. If the cell is located at e.g. the generator of R_0, there tends to be lots of Y throughout the cell and the concentration of X is the limiting factor in the production of Z. Thus more Z is produced in areas of the cell where there is more X. As before, the asymmetrical concentration of Z induces a motion towards the area that results in the production of the most Z. As the

[3] In the absence of R_0 metabolite Z is the product of only one reaction, X + Y → Z + Z. Thus, if Z is to be produced we will require some of both X and Y. Metabolites X and Y are reflexively autocatalytic, i.e. X catalyzes the production of R_0 → Y and Y catalyzes the production of R_1 → X (see reactions 1 and 2 in Table 1). As generators of R_0 and R_1 are separated spatially, it is necessary for the cell to move back and forth between the two resources if it is to maintain non-zero populations of X and Y.

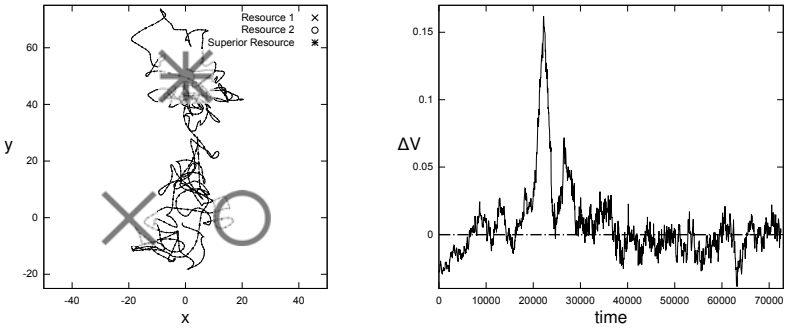


Fig. 4. Experiment 3, the protocell moves to utilize the most profitable resources, maximizing its viability

cell moves up the R_1 gradient to its generator, the concentration of Y decreases and becomes the limiting factor in the production of Z. A symmetrical process causes the cell to move back towards the original resource generator. These two process result in the oscillation of the cell between resource generators. We can again observe how the system behaves adaptively in relation to viability (Figure 3 right): when the ΔV starts to decay behavioral shifting towards the other generator inverts the tendency.

Experiment 3: Preference behavior towards better generator. First, two generators (R_1 and R_2) are presented, like in experiment 2, (see Figure 4 left) and at iteration 5000 a generator of R_0 is added at location $0, 50$. Soon after the protocell moves towards the new resource. However, as the original two resources R_1 and R_2 start to grow, they become a better quality resource and the protocell returns to them. We could interpret this behavior as an instance of action selection sensitive to viability.

4 Conclusions

We were able to observe how our model protocell behaves in relation to the conditions of long-term viability, generating instances of chemotaxis and simple (yet not trivial) examples of action selection without explicit sensors or motor and without an explicit encoding of viability conditions (as previous models of adaptive behavior assumed necessary). The adaptive nature of the behavior is shown not only in that ΔV tendencies were inverted through behavior but also because the behavior was not purely reactive nor stimulus driven. Going one step farther, the system could be interpreted as actually evaluating the value of its interactions with the environment with respect to their effect upon viability.

The model highlights the temporal aspects of the notion of viability which should be associated with tendencies of the entire situation (metabolic and environmental) rather than with regions of prohibited states. We have described the conditions under which the protocell is non-viable in the long term, and yet we see it move into those conditions and out of them in transients that are

brief enough to keep the protocell alive. This theoretical possibility (which would probably be less obvious otherwise) is highlighted by allowing a self-sustaining metabolism to move in its environment in a metabolically-regulated action. Behavior can adaptively invert the negative tendencies becoming a necessary condition for the maintenance of the system. In experiment 2, where the protocell requires two resources that are spatially separate, the long-term tendency of an unmoving cell at any point in space is certain death – no location presents a sufficient level of combined resources. However, in this environment, the cell can survive if it moves. Thus, adaptive behavior, typically conceived as something added on top of metabolism and that confers certain advantages to an already stable self-sustaining entity, turns out in this case to be an essential ingredient for the very conditions that keep the system alive. We conclude that we should remain open to seeing *agency as implicated in metabolism and metabolism as implicated in agency.*

Several measures of long-term viability could be tested instead of the one we have used here and this is a matter for further exploration, as is also the possibility of more complex behaviors enabled by more sophisticated metabolic networks and by the possibility of different forms of environmental couplings. For instance, it may be possible to explore conditions where the cell is able to perform delayed satisfaction "decisions" and other memory-related tasks, such as habituation to noxious stimuli. Such experiments may help us elucidate further the notion of viability as a temporally extended concept once the system is allowed to behave plastically.

Acknowledgements. Xabier Barandiaran is funded by Programa Nacional de Movilidad de Recursos Humanos del MEC, Plan I-D+I 2008-2011, Spain.

References

1. Ashby, W.R.: Design for a brain. J. Wiley, Chichester (1952)
2. Nolfi, S., Floreano, D.: Evolutionary Robotics. MIT Press, Cambridge (2000)
3. McFarland, D., Houston, A.: Quantitative Ethology. Pitman (1981)
4. Meyer, J., Guillot, A.: Simulation of adaptive behavior in animats: review and prospect. In: Proc. of SAB 1990, pp. 2–14. MIT Press, Cambridge (1990)
5. Beer, R.D.: The dynamics of adaptive behavior: A research program. Robotics and Autonomous Systems 20, 257–289 (1997)
6. Alexandre, G., Zhulin, I.B.: More than one way to sense chemicals. Am. Soc. Microbiol. 183 (2001)
7. Rasmussen, S., Bedau, M.A., Chen, L., Deamer, D., Krakauer, D.C., Packard, N.H., Stadler, P.F.: Protocells. MIT Press, Cambridge (2008)
8. Varela, F.J., Maturana, H.R., Uribe, R.: Autopoiesis: The organization of living systems, its characterization and a model. BioSystems 5, 187–196 (1974)
9. McMullin, B.: Thirty years of computational autopoiesis: A review. Artificial Life 10, 277–295 (2004)
10. Suzuki, K., Ikegami, T.: Emergence of protocell systems: Shapes and movements. In: Proc. of Artificial Life X Workshop (2006)
11. Egbert, M.D., Di Paolo, E.A.: Adding behavior to autopoiesis: An exploration in computational chemo-ethology. Adaptive Behavior (2009) (forthcoming)

On the Transition from Prebiotic to Proto-biological Membranes: From 'Self-assembly' to 'Self-production'

Gabriel Piedrafita[1], Fabio Mavelli[2], Federico Morán[1], and Kepa Ruiz-Mirazo[3]

[1] Dept. of Biochemistry and Molecular Biology, University Complutense of Madrid, Spain
[2] Dept. of Chemistry, University of Bari, Italy
[3] Dept. of Logic and Philosophy of Science / Biophysics Research Unit (CSIC-UPV/EHU)
University of the Basque Country, Spain
kepa.ruiz-mirazo@ehu.es

Abstract. A model is here presented to analyse how vesicles may turn into protocells that synthesize their own lipid components and the consequences that this may have on the properties of the resulting membrane (in particular, on its permeability), as well as on the overall stability of the system.

Keywords: Protocell dynamics, Gillespie algorithm, self-assembly, autopoiesis.

1 Introduction

We are engaged in an effort to understand major transitions in the development of plausible prebiotic compartments, towards proper biological membranes: i.e., membranes that can support (and be supported by) complex metabolic networks. Until recent years, most theories of the origin of life [de Duve 1991; Eigen 1992; Kauffman 1993] considered that compartmentation was a relatively late landmark, because it leads to several difficulties. In particular, it hinders free accessibility of substrates to the enclosed reaction domain. Nevertheless, at the same time, a closed membrane can have a positive effect because --for the same reason-- it should contribute to avoid the dilution or free diffusion of potentially beneficial products of those reactions, providing an aqueous environment where compound concentration levels may progressively increase.

An additional problem for the hypothesis of an early compartmentation is related to the fact that all phospholipids that constitute present day biomembranes are rather complex molecules, whose synthesis is enzymatically controlled [Peretó et al. 2004] and, thus, highly improbable in prebiotic conditions. However, other types of amphiphilic (lipid-like) compounds, like simple isoprenoids [Ourisson & Nakatani 1994] or fatty acids [Monnard & Deamer 2002], have been proposed as a more tenable starting point and have been shown to self-assemble spontaneously into stable vesicles (closed bilayers). In particular, fatty acid vesicles are being extensively investigated (for a good, recent review, see: [Morigaki & Walde 2007]) and it is being found that their dynamic properties are quite different from those of standard liposomes (i.e., phospholipid vesicles). Among other things, it has been demonstrated that they can catalyze their own formation [Walde *et al.* 1994; Bloechliger *et al.* 1998] and be

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 256–264, 2011.

grown and reproduced in *in vitro* conditions [Berclaz *et al*. 2001; Hanczyc *et al.* 2003; Chen & Szostak 2004a-b], which is very difficult to achieve with standard liposomes. Given these interesting properties, even experiments of competition/selection among different vesicle populations have been carried out [Cheng & Luisi 2003; Chen *et al*. 2004].

Another important feature of this type of vesicles is that they are much more 'leaky' -- i.e., permeable to different substances, as compared to standard liposomes. This seems quite reasonable, because it allows us to conceive the compartment of primitive protocells or vesicles as an easier barrier to cross, at least initially. Experimental confirmation of this fact has been reported by the group of Szostak [Mansy *et al*. 2008], in an interesting work showing how the permeability of vesicles changes depending on the type of lipid (or lipid mixture) used – see also [Deamer 2008]. Previous studies by Deamer and colleagues (see, e.g.: [Monnard & Deamer 2001]) had shown that the length of the hydrophobic chain of the lipid used had an important effect in the permeability of the bilayer (as it is expectable: the longer the chain, the smaller the permeability). But the series of experiments recently carried out in Szostak's lab makes clear that there are other relevant features affecting permeability, like the irregularities on the surface (caused by the different polar heads), or the fluidity and packing density of the hydrophobic tails.

All these *in vitro* experiments are assuming a scenario in which some type of prebiotic lipids are already present in the environment, studying the properties of the vesicles that they typically form by spontaneous self-assembly. Nevertheless, at some point in protocell evolution a critical transition had to occur in which self-assembling membranes became also *self-produced*. In that transition, the naturally occurring lipid (e.g., a fatty acid) would be progressively substituted by an internally synthesized different one (e.g., a phospholipid). As that conversion in the composition of the membrane takes place, its properties will also change. In particular, its permeability to the different compounds coming in and out will certainly be modified, and this can have critical effects for the stability of the whole system.

The aim of the present work is to develop a realistic model that addresses this problem, namely, the transformation of a self-assembling *vesicle* into a self-producing *protocell*, focusing on the consequences that the induced permeability changes may have in the viability of the latter. In order to do so, we make use of our platform 'ENVIRONMENT' [Mavelli & Ruiz-Mirazo 2007; Mavelli *et al.* 2008], which has been recently adapted to deal with membranes whose permeability is composition-dependent, as explained below.

2 Methods: Lipid-Composition-Dependent Permeability in ENVIRONMENT

'ENVIRONMENT' is an object-oriented (C++) platform that has been developed to simulate stochastically (by means of a Monte Carlo algorithm: the Gillespie method [Gillespie 1976; 1977]) chemically reacting systems in global non-homogeneous conditions, in which diverse aqueous and lipidic domains are defined. Our general objective is to elaborate a computational tool that allows exploring *in silico* the complex, self-organizing dynamics of systems where chemical reactions get coupled with compartment

self-assembly and diffusion/transport processes. The general features of this platform have been already described elsewhere [Mavelli & Ruiz-Mirazo 2007; Ruiz-Mirazo & Mavelli 2008; Mavelli *et al.* 2008], so here we will just briefly discuss its main novelty: i.e., the approach used to simulate solute permeability as a function of the membrane composition. A more detailed explanation can be found in the supporting material (available at: http://www.ehu.es/ias-research/ruiz-mirazo/).

In this first approximation to the problem, we shall assume that the solute diffusion coefficient of a mixed membrane D_X^{Mix} depends linearly on the membrane composition, according to the formula:

$$D_X^{Mix} = D_X^1 - \left(D_X^2 - D_X^1 \right) \chi_2^S$$

where D_X^j (j=1,2) are the diffusion coefficients of solute X across the pure membrane of amphiphiles 1 and 2, respectively, and $\chi_2^S = a_2 N_2 / \left(2 S_\mu \right)$ is the surface contribution of the second amphiphile (a_2 the area of its polar head, N_2 the total number of that amphiphile in the membrane and S_μ the total surface of the membrane -- factor 2 to account that it is a bilayer). We are aware that, in reality, things are much more complex (certainly non-linear), but the lack of adequate experimental data on this question --to our knowledge-- does not make easy a more accurate approximation.

3 Main Modelling Assumptions and Protocell Scenario

In our model, even if the initial shape of vesicles/proto-cells will be taken as spherical (for the sake of standardizing initial conditions), it is not assumed that they must stay spherical all the time, or that they divide when they double their initial size (as it is typically done). Instead, we consider that there is a relatively free relationship between volume and surface, within the following limits:

1) The actual surface of a protocell must be bigger than the theoretical spherical surface that corresponds to the actual volume at each iteration step. Otherwise the protocell will burst (in a simulated 'osmotic crisis' -- or massive water inflow).

2) The actual surface of a protocell must be smaller than the theoretical surface that corresponds to two equal spheres of half the actual volume at each iteration step. Otherwise the protocell divides, giving rise to two statistically equivalent ones.

So these are the conditions for system stability in the model: i.e., they define the range of possible states in which our protocells will not break or divide (for more details, see: Mavelli et al. 2008). In terms of Φ, the 'reduced surface' of the system (i.e., the ratio between the actual surface S_μ and the surface of an ideal sphere of volume V_{core} : $\Phi = S_\mu / \sqrt[3]{36\pi V_{core}^2}$) those conditions can be expressed as $1 \le \Phi \le \sqrt[3]{2}$. Besides, if one takes into account that the membrane is a relatively elastic structure, two additional parameters can be introduced as follows:

$$1 - \varepsilon \le \Phi \le (1 + \eta) \sqrt[3]{2}$$

where ε and η are the burst and fission tolerance, respectively. Although these two parameters may change as functions of the membrane composition

[Ruiz-Mirazo & Mavelli 2008], in all simulation runs reported below they were fixed equal to 0.21 and 0.1 (respectively), so the actual stability range becomes $0.79 \leq \Phi \leq 1.386$.

Under these general conditions and modelling assumptions, in the present work we explored a scenario in which already formed vesicles become protocells thanks to a series of internal reactions (an autocatalytic cycle that represents a rather elementary proto-metabolism, as depicted in Fig. 1) that transform a precursor amphiphile (l) --a fatty acid, for instance, making up the initial membrane--, into a more complex lipid molecule (L) --say, a phospholipid--. The critical aggregation concentration (cac) or relative solubility of this new lipid is much lower than that of the precursor (as when comparing real fatty acids and phospholipids) so most of what is internally produced is rapidly incorporated to the growing membrane, whose composition changes accordingly.

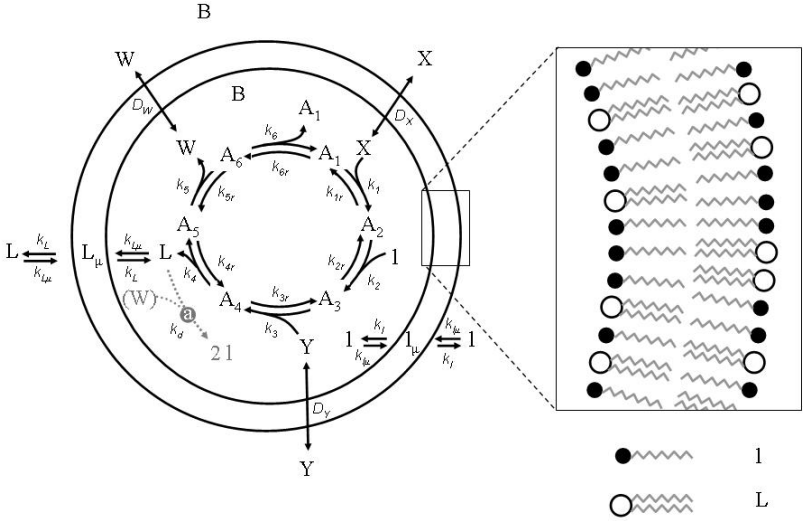


Fig. 1. Reaction scheme through which a single chained lipid (l), with the addition of two externally fed molecules (X and Y) and the contribution of a series of metabolites (catalysts A_i; i=1-6, at least one of whose initial concentration should be non-zero) is transformed into a more complex, double chained lipid (L), producing some waste (W) or leaving group, as well as one of the intermediary metabolites (to make the cycle properly autocatalytic – Ganti 2002). (a) The spontaneous decay of L into l (or 2l), with or without the intervention of W, was introduced to find conditions under which the final stationary state of the membrane is not pure. B is a non-reactive species (osmotic buffer), which is important to include for general stability reasons.

As a result of this transformation the membrane is expected to become less permeable to the different chemical species that can cross it (the nutrient molecules, X and Y, and the waste product W). And, similarly to what occurred in our previous work [Mavelli & Ruiz-Mirazo 2007; Ruiz-Mirazo & Mavelli 2008], the accumulation of a non-functional compound, W, within the boundaries of the system is a big threat to its stability. Therefore, if the lipid conversion (from l to L) is complete and, as a consequence, the permeability --i.e., rate of release-- of W decreases significantly,

there are high chances that the system will undergo an 'osmotic crisis'. In order to counterbalance the strength of the autocatalytic cycle in this sense, we introduced different types of decay processes (back from L to l), searching for stationary states in which the membrane composition turns out to be mixed -- see more details below.

4 Results

As illustrated in Fig. 2, there are three main possible outcomes in the time evolution of the protocells in our model: they can get into a self-reproducing regime (subsequent growth and splitting processes), find --or tend asymptotically to-- a self-maintaining state, or suffer an osmotic burst. This depends on a series of factors, among which we have chosen to study here, in particular, the following two: changes in the permeability to the waste (P_w) and rate of decay of L into l (k_d). Almost all the remaining variables, initial size of the vesicle, kinetic constants, values of initial concentrations... were kept the same in all reported simulation runs (further details: http://www.ehu.es/ias-research/ruiz-mirazo/Suppl_info_Piedrafita_etal_2009.pdf).

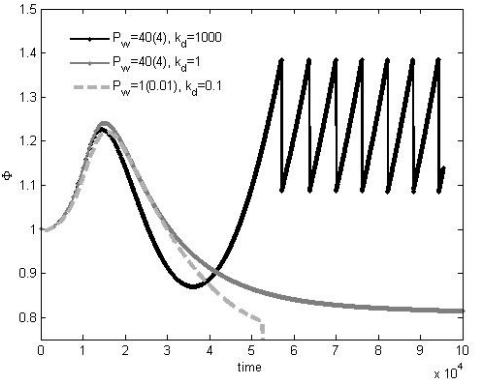


Fig. 2. Time course of the 'reduced surface' (Φ) between its two critical values (0.79 -- 1.386) for three different cases: self-reproduction (in black), self-assembly (dark grey) and osmotic burst (light grey). The graph also shows the two main parameters that determined the outcome of our simulations in this work: the permeability coefficient ($x10^{-8}$ cm/s) to the waste for each of the lipids (in brackets the less permeable one) and the constant of decay of L into l (in all these cases: L+W → 2l).

First of all we analysed the situation when there is no decay at all. In that case, the forced conversion of l into L involves the progressive accumulation of W within the protocell, until it typically breaks down. This can only be overcome if bigger values of the permeability are set (allowing for a faster waste release) or if the overall rate of the internal production cycle, for whatever reason, starts going down, eventually ceasing to operate (for instance, due to the limited amount of available l, as in Fig. 3A). More interesting dynamics were found when some type of spontaneous decay of the complex lipid was included in the reaction scheme so, at the same time as L is

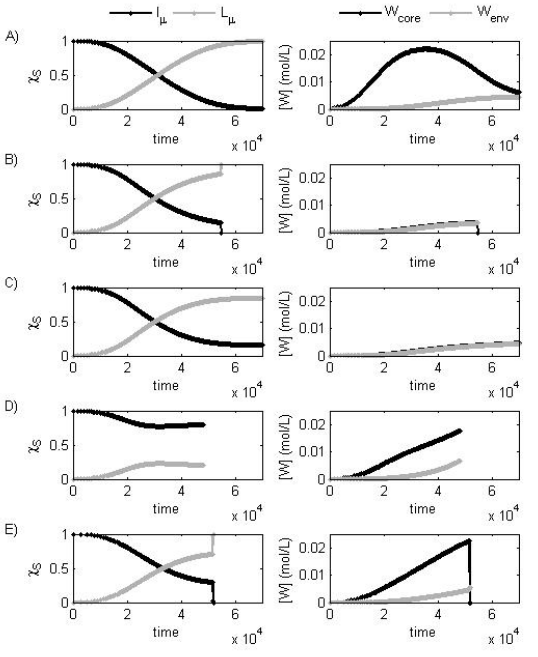


Fig. 3. Time evolution of the relative membrane surface contribution of l and L (χ_{Sl} and χ_{SL}, respectively -- left column) and of the external and internal waste concentration (W_{env} and W_{core}, respectively -- right column). A) No spontaneous decay of L, with P_x, P_y = 24 x10^{-8} (cm s^{-1}), and P_w = 0.25x10^{-8} (cm s^{-1}). Although there is no sudden osmotic crisis, the internal cycle progressively ceases to operate, leading to the eventual equilibration between W_{core} and W_{env}. Then, in B) and C) a decay process consuming L and W is included, giving rise to a single molecule of l (B) or two molecules of l (C), with different outcomes. For both simulations k_d = 10 (M^{-2} s^{-1}) and P_x, P_y, P_w = 40(4) x10^{-8} (cm s^{-1}). Finally, we considered a decay process that produces 2l without consumption of W (D and E). $P_{x/y}$ = 24x10^{-8} (cm s^{-1}) and P_w = 1(0.01) x10^{-8} (cm s^{-1}). The difference now is in the rate of decay (k_d): equal to 1 (D) or 0.1 (M^{-1} s^{-1}) (E).

produced, part of it converts back, irreversibly, into l. If the decay reaction involves also the waste or leaving group (e.g., L + W → l), the system could, at the same time, cut down its excessive production. However, this does not ensure the stability of the protocell, because the total amount of lipid (L+l) internally produced may not be enough for a membrane that needs to grow in parallel to proto-metabolism (Fig. 3B).

Alternatively, the stoichiometry may favour a rapid enough growth of the membrane if the overall amount of lipid molecules synthesized increases, with or without the participation of W (e.g., L (+ W) → 2l), assuming that either X or Y is acting as lipid precursor structurally close to l (Fig. 3C-D). Nevertheless, kinetic parameters, like the rate of decay of L (k_d), keep an important influence on the evolution of the system, particularly when it is close to a critical point, as shown in Figs. 3D-E. If k_d is relatively high it means that the synthesized complex lipid L decays rapidly into l, so the conversion of the membrane does not effectively occur, and its permeability

remains quite high (Fig. 3D). In contrast, if k_d is smaller, the membrane is transformed into a more impermeable bilayer, with higher content of L, which can be lethal (Fig. 3E).

5 Discussion and Final Remarks

In this paper we have assumed that the transition from self-assembling compartments, made of a naturally occurring precursor lipid, to self-producing or *autopoietic* protocells [Varela et al. 1974], with a membrane made of an internally synthesized, more complex lipid, might take place through a robust, Ganti-type autocatalytic cycle [Ganti 2002] that helps transform one into the other. We have shown how such a simple cycle-reaction network may support that kind of conversion and, depending on the initial conditions and kinetic parameters chosen, different types of membrane (mixed or pure) are obtained. However, this kind of reaction scheme is, at the same time, perhaps too powerful, in the sense that it seems to lead, sooner or later, to the complete conversion of the membrane; i.e., apparently, it does not favour stationary states of coexistence or mixture of the two membrane lipids -- although a wider exploration of parameter space is being carried out to check this.

The reason why such a scenario would be more interesting to study is because, as we already mentioned, the complete transformation of the membrane may not be the best strategy in the development of functionalized compartments, given their expected higher stability and lower permeability. So, in order to counterbalance the strength of the internal proto-metabolic cycle, some spontaneous decay of the complex lipid into its precursor was introduced. Nevertheless, it was not easy to drive the system to a mixed-composition stationary state: either L or l eventually displaced the other (sometimes after a considerably long transition period in which both did coexist). Again, further work and longer simulation runs need to be performed.

Anyhow, if the present state of affairs is confirmed, we may be led to the conclusion that either (i) this type of reaction scheme is not the most suitable for the transition we are trying to model here (from 'self-assembly' to 'self-production'), so different alternatives ought to be proposed and carefully analysed; or (ii) the type of reaction scheme would be correct, but needs to involve other compounds, not just lipids. This second possibility would be coherent, for instance, with our previous claims [Ruiz-Mirazo & Moreno 2004; Ruiz-Mirazo & Mavelli 2007; 2008] that peptides are crucial for the development of functional compartments. In other words, the suggestion would be that it may not be possible to seal precursor, leaky vesicles through the synthesis of new, more impermeable types of lipid, unless the system is capable to synthesize, in parallel, other compounds (e.g., peptides) by means of which it can regulate the transport of substances in and out, as well as the potential osmotic imbalances that can --and surely will-- be created in its relation with the environment.

Acknowledgements. Gabriel Piedrafita, who holds an *FPU* - PhD scholarhip from the Spanish Ministry of Science, and Federico Morán acknowledge support from the research project BFU2006-01951 (MICINN, Spain). Kepa Ruiz-Mirazo, a *Ramón y Cajal* research fellow, in turn, acknowledges support from grants IT-250-07 (Basque Government) and FFI2008-06348-C01/02/FISO (MICINN, Spain).

References

1. Berclaz, N., Mueller, M., Walde, P., Luisi, P.L.: Growth and Transformation of Vesicles Studied by Ferritin Labeling and Cryotransmission Electron Microscopy. J. Phys. Chem. B 105, 1056–1064 (2001a)
2. Bloechliger, E., Blocher, M., Walde, P., Luisi, P.L.: Matrix Effect in the Size Distribution of Fatty Acid Vesicles. J. Phys. Chem. 102, 10383–10390 (1998)
3. Chen, I.A., Roberts, R.W., Szostak, J.W.: The Emergence of Competition Between Model Protocells. Science 305, 1474–1476 (2004)
4. Chen, I.A., Szostak, J.W.: A Kinetic Study of the Growth of Fatty Acid Vesicles. Biophys. J. 87, 988–998 (2004a)
5. Chen, I.A., Szostak, J.W.: Membrane growth can generate a transmembrane pH gradient in fatty acid vesicle. Proc. Nat. Ac. Sci. USA 101(21), 7965–7970 (2004b)
6. Cheng, Z., Luisi, P.L.: Coexistence and mutual competion of vesicles with different size distributions. J. Phys. Chem. B 107(39), 10940–10945 (2003)
7. Deamer, D.W.: Origins of life: how leaky were primitive cells? Nature 454, 37–38 (2008)
8. de Duve, C.: Blueprint for a cell: the nature and origin of life. Neil Patterson Publishers, Burlington (1991)
9. Eigen, M., Winkler-Oswatitsch, R.: Steps towards life: A perspective on evolution. Oxford Univ. Press, New York (1992)
10. Ganti, T.: On the early evolutionary origin of biological periodicity. Cell Biol. Int. 26, 729–735 (2002)
11. Gillespie, D.T.: A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Reactions. J. Comput. Phys. 22, 403–434 (1976)
12. Gillespie, D.T.: Exact Stochastic Simulation of Coupled Chemical Reactions. J. Phys. Chem. 81, 2340–2369 (1977)
13. Hanczyc, M.M., Fujikawa, S.M., Szostak, J.W.: Experimental Models of Primitive Cellular Compartments: Encapsulation, Growth, and Division. Science 302, 618–621 (2003)
14. Kauffman, S.: The origins of order: Self-organization and selection in evolution. Oxford Univ. Press, Oxford (1993)
15. Mansy, S., et al.: Template directed synthesis of a genetic polymer in a model protocell. Nature 454, 122–126 (2008)
16. Mavelli, F., Ruiz-Mirazo, K.: Stochastic simulations of minimal self-reproducing cellular systems. Phil. Trans. Royal. Society of London B 362(1486), 1789–1802 (2007)
17. Mavelli, F., Lerario, M., Ruiz-Mirazo, K.: 'ENVIRONMENT': a stochastic simulation platform to study protocell dynamics. In: Arabnia, H.R., et al. (eds.) Proceedings of BIOCOMP 2008, vol. II, pp. 934–941. CSREA Press (2008)
18. Monnard, P.-A., Deamer, D.W.: Nutrient uptake by protocells: a liposome model system. Origins of Life and Evolution of the Biosphere 31, 147–155 (2001)
19. Monnard, P.-A., Deamer, D.W.: Membrane Self-Assembly Processes: Steps Toward the First Cellular Life. Anat. Rec. 268, 196–207 (2002)
20. Morigaki, K., Walde, P.: Fatty acid vesicles. Cur. Opin. Coll. & Interface Science 12, 75–80 (2007)
21. Ourisson, G., Nakatani, Y.: The terpenoid theory of the origin of cellular life: the evolution of terpenoids to cholesterol. Chem. & Biol. 1, 11–23 (1994)
22. Peretó, J., et al.: Ancestral lipid biosynthesis and early membrane evolution. TIBS 29(9), 469–477 (2004)

23. Ruiz-Mirazo, K., Moreno, A.: Basic autonomy as a fundamental step in the synthesis of life. Artificial Life 10(3), 235–259 (2004)
24. Ruiz-Mirazo, K., Mavelli, F.: Simulation Model for Functionalized Vesicles: Lipid-Peptide Integration in Minimal Protocells. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 32–41. Springer, Heidelberg (2007)
25. Ruiz-Mirazo, K., Mavelli, F.: On the way towards 'basic autonomous agents': stochastic simulations of minimal lipid-peptide cells. BioSystems 91, 374–387 (2008)
26. Varela, F.J., Maturana, H., Uribe, R.: Autopoiesis: The Organization of Living Systems, its characterization and a model. BioSystems 5, 187–196 (1974)
27. Walde, P., Wick, R., Fresta, M., Mangone, M., Luisi, P.L.: Autopoietic Self-Reproduction of Fatty Acid Vesicles. J. Am. Chem. Soc. 116, 11649–11654 (1994)

SimSoup: Artificial Chemistry Meets Pauling

Chris Gordon-Smith

SimSoup
c.gordonsmith@gmail.com
http://www.simsoup.info

Abstract. Theories of the Origin of Life can be categorised as 'template replication first' and 'metabolism first'. A key question for metabolism first theories is the mechanism for transfer of inherited information. Earlier work presented a mechanism based on catalytic cycles, along with supporting results from the SimSoup artificial chemistry simulator[1]. The current paper presents an enhanced SimSoup model that is closer to real chemistry and more open ended. Molecules and the types of Interactions between them are constructed by the model itself using simple rules based on valence theory. Results of a preliminary run of the model are presented. Most of the Molecules produced are of a few simple types with low molecular weight. There is a 'long tail' of many low frequency Molecules, many of which are more complex with high molecular weight.

1 Introduction

1.1 Motivation

There is currently no complete and generally accepted explanation of the Origin of Life. Evolution requires inherited information to be passed from parent to offspring; contemporary organisms do this using template replicating molecules (eg DNA). However, the mechanism requires the assistance of highly evolved proteins, which would not have been present in the prebiotic world. Metabolic theories of the Origin of Life propose that early organisms were metabolic systems that transmitted inherited information without the use of template replicators.

The motivation for the SimSoup project is to show that a metabolic network has sufficient information carrying properties to enable evolution to begin without the assistance of highly evolved molecules such as proteins or DNA.

1.2 Conceptual Background

The conceptual background for SimSoup includes the following:-

- The metabolic theories of Aleksandr Oparin (Oparin [8]), Stuart Kauffman (Kauffman [7]), Freeman Dyson (Dyson [2]), Fernando and Rowe (Fernando and Rowe [3]), and more specifically the Lipid World and the GARD model of Doron Lancet's group (Segré et al. [10]).

[1] SimSoup has been implemented as a computer simulation. The C++ source code is available at Ref [11].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 265–272, 2011.

- Network theory, particularly the work of Sanjay Jain and Sandeep Krishna (Jain and Krishna [6]).
- Günter Wächtershäuser's chemo-autotrophic Iron-Sulphur World (Wächtershäuser [13] [14]).
- Chemical bond theory, as presented in Pauling [9].

2 The SimSoup Artificial Chemistry Model

2.1 SimSoup as a Network Simulator

The following summarises the model presented in Gordon-Smith [4] and [5]:

- **Chemical Network:** A network is defined in which the nodes are Molecule Types and connections are Interaction Types. Molecule Types have Mass and Potential Energy, but no other properties. Interaction Types take three forms: Construction ($A + B \rightarrow C$), Fission ($A \rightarrow B + C$), and Transformation ($A \rightarrow B$). Each Interaction Type must conserve Mass, and has an Activated Complex Energy that is used along with the Potential Energies of the Molecule Types to determine the forward and reverse Rate Constants. This makes the model thermodynamically realistic.
- **Network Dynamics:** Interactions take place between Molecules in a well stirred Reactor. Each bimolecular Interaction Type (Construction) occurs at a rate equal to the product of the Rate Constant and the concentrations of the two Reactants. Each unimolecular Interaction Type (Fission or Transformation) takes place at a rate equal to the product of the Rate Constant and the concentration of the (single) Reactant.
- **Compound Interactions:** Interaction Types can be combined in ways that have catalytic properties. For example, $A + X \rightarrow I$ followed by $I + B \rightarrow J$ and finally $J \rightarrow X + C$ together make up a Compound Interaction with overall scheme $A + B \xrightarrow{X} C$, with X operating as a catalyst, and I and J as intermediates.
- **Trackers and Cycle Detection:** A Tracker is an object that can be attached to a Molecule. As Molecules take part in Interactions, the Trackers are passed from Reactant Molecule to Product Molecule. This enables cycles to be detected and monitored.
- **Data Series Plots:** These show the real-time behaviour of a range of variables that are monitored as the simulation runs.
- **Manhattan Plot:** This shows the variability in the composition of the material in the Reactor over time.

2.2 SimSoup Extended as a Network Explorer

A More Open-Ended Approach - Adding Molecular Structure: In the earlier model of Section 2.1, the chemical network is defined as part of each model scenario. This limits the scope for the system to generate novelty. To deal with this, SimSoup has been enhanced as follows:

- Each Molecule Type has a structure.
- Molecules can Join or Split to form Molecules of different types.
- Joining and Splitting occur in ways that depend on the structural properties of the Reactant(s).

The rules that determine the structure of Molecules and the way in which they Join and Split are designed to be analogous to real chemistry, while keeping the model conceptually simple and computationally tractable. The key objective is to produce an *open ended model in which the opportunities for novelty are similar to those that exist in real chemistry.*

Overview of Molecular Structure in SimSoup: Molecules are two dimensional rigid structures built from Atoms that are bonded together such that they occupy fixed positions on a square 'Board' (similar to a chess board). Each square may contain at most one Atom. Bond angles are always 90° or 180°, and bond lengths are all equal. Atoms bond together in a way that is broadly consistent with valence bond theory in real chemistry, and with comparable bond energies.

Atom Types: SimSoup Atoms are of different types, with an Atom Type being characterised by Mass, plus the following properties:-

- $N_{ElecFullShell}$: Number of electrons to fill the outer electron shell.
- $N_{ValElec}$: Number of (valence) electrons in the outer shell.

This is by analogy with real chemistry, in which the ability of atoms to bond depends on shell structure. Atoms with full shells include stable noble gases, which rarely bond to other atoms.

A SimSoup Atom can bond to other Atoms, with the maximum number of bonds that can be supported being $N_{PossBond} = N_{ElecFullShell} - N_{ValElec}$.[2]

$N_{ElecFullShell}$ and $N_{ValElec}$ (and so $N_{PossBond}$) for SimSoup Atom Types are based on real chemistry. The Atom Types modelled are Hydrogen, Carbon, Nitrogen and Oxygen, with values of 1,4,3 and 2 respectively for $N_{PossBond}$.

Bonds, Bond Order and Bond Energy: SimSoup models single, double and triple order (covalent) bonds, so that each of the following is possible: O—H, C═O, O═C═O, C≡N. Bond Energies are the averages for bonds in real chemistry, taken from Pauling [9] and Atkins [1].

Example Molecular Structures: Figure 1 shows some simple Molecules that have been produced by SimSoup, and the real molecules to which they are analogous. The analogy is not perfect. For example, real methane is three dimensional and tetrahedral with bond angle 109.5°, and the bond angle in real water is 104.5°, rather than 90°. In addition, there are two forms of 'water' in SimSoup, the second being linear: H—O—H. The correspondence is closer for acetylene and hydrogen cyanide, which are linear both in real chemistry and in SimSoup.

[2] Real atoms sometimes do not have such a simple limit on the number of possible bonds. The five bonds in phosphorous pentachloride provide an example.

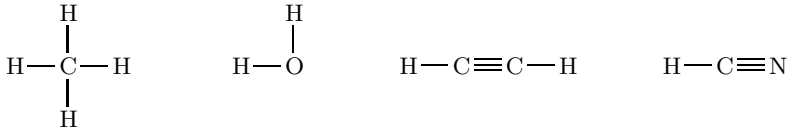


Fig. 1. Example SimSoup Molecules

Joining Molecular Structures: Whenever a Molecule Type becomes actualised in the Reactor (the number of Molecules of the type present changes from zero to one), it is checked against all the other actualised Molecule Types to determine, in each case, whether a join is possible.

Two Molecules can Join (in a Construction Interaction) subject to the following rules:

- Two Atoms cannot occupy the same Board position in the joined Molecule.
- No Atom can have more than $N_{PossBond}$ bonds.
- Bond Energies are the averages for the corresponding real chemistry bonds.
- When a pair of Atoms is considered for bonding, the highest order bond that is feasible (within the constraint on $N_{PossBond}$ for each Atom) is formed.
- Molecules Join in a way that maximises total Bond Energy. If multiple Joins with the same (highest) energy are possible, the first one found is selected.

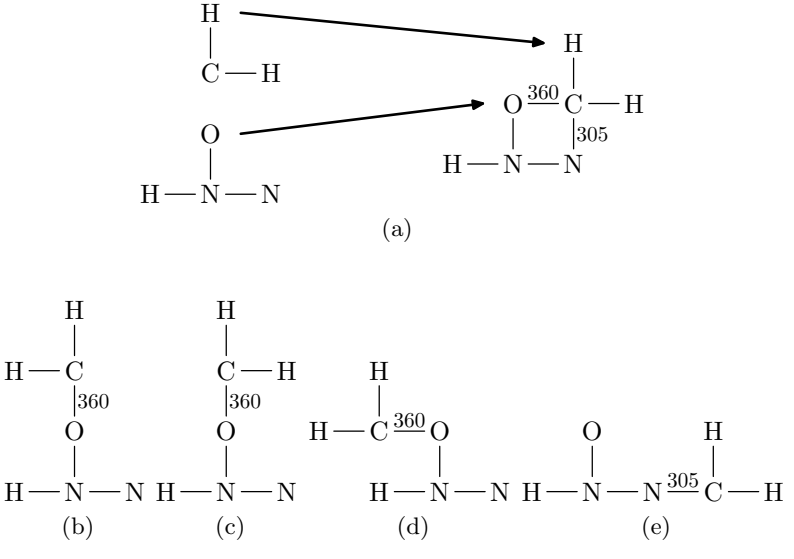


Fig. 2. (a) Example Construction showing Reactants and most stable (highest total bond energy) Product. The new bonds created are labelled with the bond energy (kJ mol^{-1}) in each case. (b) to (e) show some of the possible alternative Products.

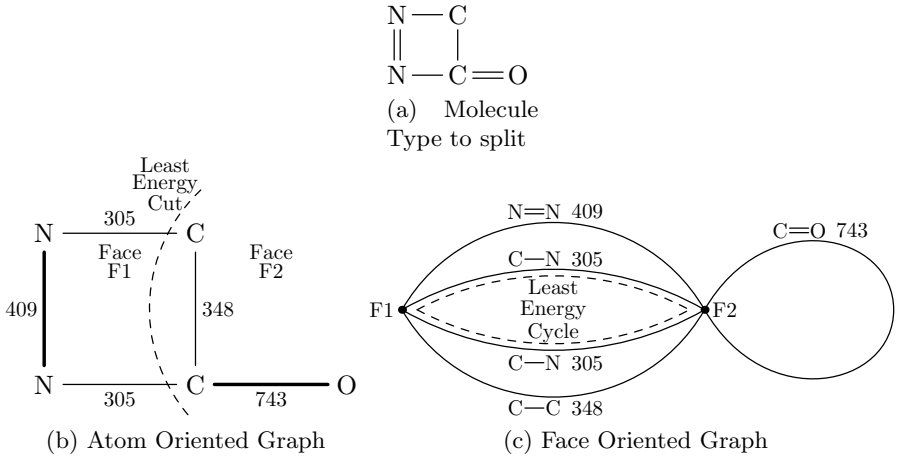


Fig. 3. Illustration of the SimSoup Splitting algorithm

Figure 2 shows an example Interaction Type (a Construction) in which two Molecules Join to form a single Product. The figure also shows several (but not all) of the 'rejected' Products with lower total bond energies. The Product in Figure 2(a) is the only one that entails the formation of two new Bonds.

Splitting Molecular Structures: When a Molecule Type is actualised for the first time, a Fission Interaction Type is determined. This defines the two Products into which a Molecule of the type can Split. The Products are identified by finding a set of Bonds that, if broken, would Split the Molecule into two unconnected parts. Where there are multiple ways in which the Molecule can be split, as is usually the case, the split identified is one that entails breaking bonds with the lowest possible total energy.

The splitting algorithm works on Molecules of arbitrary complexity. It uses techniques of graph theory[3], and is illustrated in Figure 3. The major steps are:

- Represent the Molecule Type to be split as an 'Atom Oriented Graph'.
- Transform this into a 'Face Oriented Graph'.
- Use the Dijkstra shortest paths algorithm to find a least energy cyclic path in the Face Oriented Graph. The path edges identify the Bonds to break.

The remainder of this section explains the algorithm.

Representing The Molecule as an Atom Oriented Graph: Each node in this graph represents an Atom in the Molecule to be split, and each edge represents the (single or multiple) Bond between the Atoms that the edge connects. Each edge has a weight indicating Bond Energy. Figure 3(b) shows the Atom Oriented Graph for the Molecule Type in Figure 3(a). The thicker edges represent double bonds. The numbers by the edges are the Bond Energies (kJ mol^{-1}).

[3] The SimSoup implementation makes use of the Boost Graph Library code.

The Atom Oriented Graph is always planar in SimSoup[4]; that is, it can be embedded in a flat plane in such a way that the edges intersect only at their endpoints. A planar embedding divides the plane into regions or *faces*. One face is unbounded or *external*[5]. In Figure 3(b), face F2 is the external face.

It can be shown that for any single component planar graph, *separating the graph into two components requires a 'cut' that starts on one face, crosses one or more edges, and returns to the starting face.* The task is to find a 'least energy cut' that breaks the bonds with the least total energy.

It can be seen from Figure 3(b) that the structure will split with the breaking of two different bonds. Although the least energy cut crosses two edges, the energy is less than for the alternative cut that starts and finishes on face F2 and removes only the one edge for the C═O double bond.

The Role and Usage of the Face Oriented Graph: For a Molecule of arbitrary complexity there will typically be many faces and many possible cuts, and it is not immediately obvious how to find the least energy cut.

The problem can be dealt with by transforming the Atom Oriented Graph to a form that enables the well known Dijkstra shortest paths algorithm to be used. This is the purpose of the transformation to the Face Oriented Graph.

The vertices in the Face Oriented Graph correspond to the faces in the Atom Oriented Graph. The edges correspond to the edges in the Atom Oriented Graph, but the vertices (or vertex) to which an edge is connected correspond(s) to the face(s) on either side of the edge in the Atom Oriented Graph. For example, the C—C bond in Figure 3(b) separates faces F1 and F2; it corresponds to the lowest edge in Figure 3(c) that joins vertices F1 and F2.

The problem of finding a least energy cut of the Atom Oriented Graph is equivalent to finding a *cyclic path through the Face Oriented Graph whose edges have a total Bond Energy less than or equal to that for any other cyclic path.*

SimSoup uses Dijkstra's shortest path algorithm to find such a path, with the energy of each edge being interpreted as its 'length'. Each edge in the Face Oriented Graph is considered in turn, and Dijkstra's algorithm is used to find a (different) least energy path (if any) between the edge's endpoints, thereby completing a cycle. The cycle 'length' is the energy of the edge plus the total energy of the (different) shortest path. By considering each edge in this way, a 'shortest' cycle with the lowest total energy is found. The edges in the cycle identify the bonds to break to split the Molecule Type.

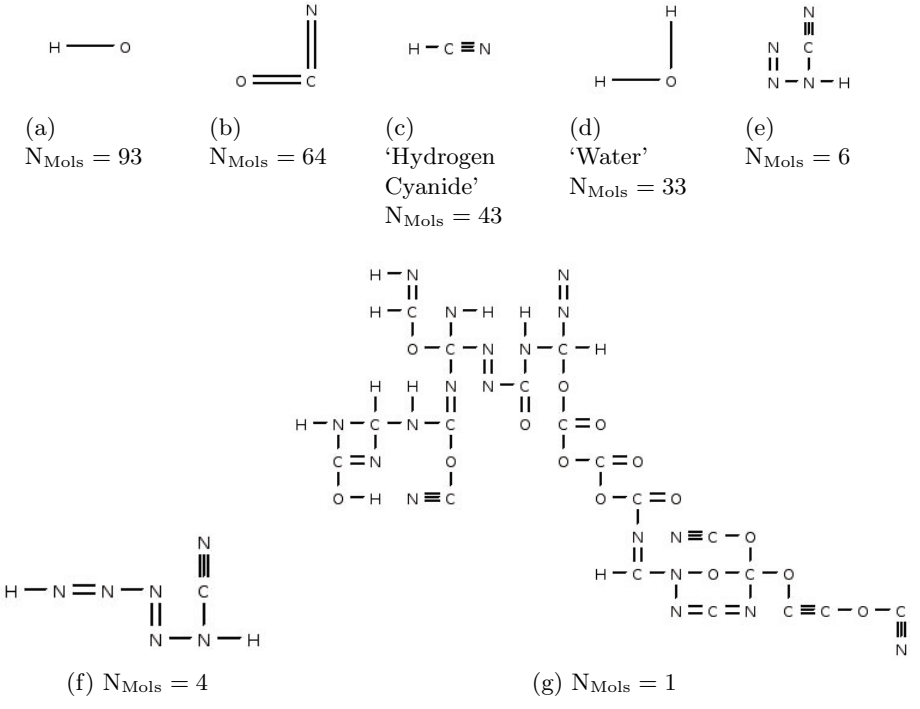
3 Preliminary Results

Some preliminary output[6] is included in Figure 4. The figure shows some Molecule Types present in the SimSoup Reactor at the end of a run. The run was setup such

[4] This is a result of the 'Board' layout of Molecules and the fact that Bonds are always with Atoms in neighbouring board squares.

[5] The external face has no particular significance. If the graph were embedded in the surface of a sphere there would be no external face.

[6] Results were produced using SimSoup revision 97, source code available at Ref [11].



(a) $N_{Mols} = 93$

(b) $N_{Mols} = 64$

(c) 'Hydrogen Cyanide' $N_{Mols} = 43$

(d) 'Water' $N_{Mols} = 33$

(e) $N_{Mols} = 6$

(f) $N_{Mols} = 4$

(g) $N_{Mols} = 1$

Fig. 4. Molecule Types in a preliminary model run, and number (N_{Mols}) in each case

that 10 Atoms of (each of) Hydrogen, Carbon, Oxygen and Nitrogen were added at intervals of 100 timesteps. After 5000 timesteps, there were 74 Molecule Types actualised in the Reactor. This was about 1% of the total number of Molecule Types that had been actualised during the run. The total number of Molecule Types discovered was 33383, not all of which had been actualised.

Of the 74 Molecule Types present, a few of the 11 most frequent are shown (Figure 4(a) to 4(f)), along with the number of Molecules (N_{Mols}) in each case. For the other 63 Molecule Types, only one of each was present. Several of these were larger Molecules with more than 15 Atoms. Figure 4(g) shows an example with 60 Atoms.

4 Conclusions and Prospects

4.1 Conclusions

The SimSoup model represents chemistry at the level of elementary reactions (unimolecular and bimolecular), with catalysis as a property of the network. The model has been extended to represent molecular structure in a way that is

analogous to real chemistry.[7] This makes it capable of exploring the behaviour of a vast space of (artificial) metabolic networks in a very open ended way.

Output from a preliminary model run has been presented. Most of the Molecules produced are small and of only a few types. The remaining Molecules exist in low numbers, but are more diverse and typically have a transient existence. At the end of the run, there were 74 Molecule Types present in the Reactor, but in excess of 7000 different Molecule Types had been actualised during the run.

4.2 Prospects

Possible avenues for future investigation using SimSoup include:

- High activity networks and the role played by catalytic cycles.
- The ability of such networks to 'remember' perturbations and thereby carry inherited information.
- The role of molecular structure in (non template based) inheritance.

References

1. Atkins, P.: Physical Chemistry. Oxford University Press, Oxford (2006)
2. Dyson, F.: Origin Of Life. Cambridge University Press, Cambridge (1999)
3. Fernando, C., Rowe, J.: Natural selection in chemical evolution. Journal of Theoretical Biology 247, 152–167 (2007)
4. Gordon-Smith, C.: SimSoup: An Artificial Chemistry Model for Investigation of the Evolution of Metabolic Networks. In: ECAL 2005, Artificial Chemistry Workshop Paper (2005), http://www.simsoup.info/Publications.html
5. Gordon-Smith, C.: Evolution Without Smart Molecules. In: ECAL 2007, Extending the Darwinian Framework Workshop Paper (2007), http://www.simsoup.info/Publications.html
6. Jain, S., Krishna, S.: Autocatalytic Sets and the Growth of Complexity in an Evolutionary Model, http://arxiv.org/abs/adap-org/9809003
7. Kauffman, S.A.: The Origins Of Order. Oxford University Press, Oxford (1993)
8. Oparin, A.I.: The Origin Of Life On The Earth. Oliver And Boyd (1957)
9. Pauling, L.: The Nature Of The Chemical Bond. Cornell University Press (1960)
10. Segré, D., Dafna, B., Deamer, D., Lancet, D.: The Lipid World. Origins Life Evol. Biosphere 31, 119–145 (2001)
11. SimSoup Source Code, http://code.google.com/p/simsoup/
12. Szathmáry, E.: The origin of replicators and reproducers. Phil. Trans. R. Soc. 361, 1761–1776 (2006)
13. Wächtershäuser, G.: Evolution of the first metabolic cycles. Proc. Natl. Acad. Sci. USA 87, 200–204 (1990)
14. Wächtershäuser, G.: The Origin of Life and its Methodological Challenge. J. Theor. Biol. 187, 483–494 (1997)

[7] Resonance and electronegativity are not modelled. Doing so could be considered, but is not essential for the current objective of making the model more open ended.

Elongation Control in an Algorithmic Chemistry

Thomas Meyer[1], Lidia Yamamoto[1],
Wolfgang Banzhaf[2], and Christian Tschudin[1]

[1] University of Basel, CH-4056 Basel, Switzerland
{th.meyer,lidia.yamamoto,christian.tschudin}@unibas.ch
http://cn.cs.unibas.ch
[2] Memorial University of Newfoundland, St. John's, NL, A1B 3X5, Canada
banzhaf@cs.mun.ca
http://www.cs.mun.ca/~banzhaf/

Abstract. Algorithmic chemistries intended as computation models seldom model energy. This could partly explain some undesirable phenomena such as unlimited elongation of strings in these chemistries, in contrast to nature where polymerization tends to be unfavored. In this paper, we show that a simple yet sufficiently accurate energy model can efficiently steer resource usage, in particular for the case of elongation control. A string chemistry is constructed on purpose to make strings grow arbitrarily large. Simulation results show that the addition of energy control alone is able to keep the molecules within reasonable length bounds, even without mass conservation, and without explicit length thresholds. A narrow energy range is detected where the system neither stays inert nor grows unbounded. At this operating point, interesting phenomena often emerge, such as clusters of autocatalytic molecules, which seem to cooperate.

1 Introduction

Algorithmic Chemistries [1,2] are artificial chemistries where algorithms or complex functions emerge as the outcome of stochastic interactions among simple molecules. The role of energy in these chemistries has not been sufficiently emphasized so far. Yet energy plays a crucial role in all chemical processes related to life. Living organisms need a permanent intake of energy in order to drive essential chemical reactions, which otherwise would rarely occur spontaneously. At the same time, energy limitations are important selection factors in evolution.

This paper illustrates one of the potential roles of energy in algorithmic chemistries: that of controlling resource usage in a natural way, in particular, to restrict the lengths of polymers in a string-based artificial chemistry to reasonable bounds, without resorting to arbitrary thresholds.

The paper is structured as follows: Section 2 briefly describes the energy framework used. Section 3 shows elongation control in an evolution scenario with a population of artificial cells based on a constructive chemistry that is particularly aggressive in elongating. The emergence of autocatalytic clusters is observed for a range of energy injection rates. Discussions and conclusions follow in Sects. 4 and 5.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 273–280, 2011.

2 Energy Framework

Most algorithmic chemistries operate at the microscopic level of individual molecular collisions, since the outcome of the reaction is computed from the information carried within the molecules [1,2,3]. Moreover, many algorithmic chemistries are constructive [1,3,4], producing new molecular species all the time. This is in contrast to many simulations of real chemistry, where the molecular species and their reactions are known beforehand, and where a deterministic approach based on ODEs (ordinary differential equations) is often sufficient.

We propose an energy framework aimed specifically at algorithmic chemistries, especially constructive ones. It provides a microscopic level stochastic simulation of chemical reactions in a well-stirred vessel under energy constraints, such that the qualitative behavior is similar to real chemistry, yet still sufficiently abstract to incur an acceptable computational cost.

The framework is divided into three steps: collision, reaction, and energy balancing. The first step selects molecules for collision using existing algorithms such as [5,6].

The second step (reaction) decides whether a reaction will be effective or elastic. This step is based on the fact that the kinetic rate coefficient for a given chemical reaction stems from the activation energy E_a necessary for a reaction to take place, according to the Arrhenius equation: $k = Ae^{-\frac{E_a}{RT}}$ for a reaction such as $X + Y \xrightarrow{k} Z$, where A and R are constants, and T is the absolute temperature. Since the framework operates at the microscopic level, the quantity E_a must be rescaled to a single reaction, obtaining ε_a. Moreover, the kinetic energy ε_k of the colliding reactants must be modeled with low computation penalty. We do this by tracking the overall kinetic energy E_k in the vessel of N molecules rather than the velocity of individual molecules and by drawing ε_k from an exponential distribution, based on results from statistical mechanics [7]: $\varepsilon_k \sim \mathrm{Exp}(E_k/N)$. The reaction is effective if $\varepsilon_k > \varepsilon_a$. It can be easily demonstrated that this procedure leads to Arrhenius behavior at the macroscopic level (mathematical derivation omitted for conciseness).

The third step implements energy conservation, i.e. the first law of thermodynamics, by making sure the energy amounts are properly balanced before and after the reaction. Figure 1 shows a typical energy diagram for a reaction. Energy is conserved as it proceeds from reactants (left side) to products (right side): $\varepsilon_k + \varepsilon_{p,i} = \varepsilon_o + \varepsilon_{p,o}$. Since $\varepsilon_{p,o} < \varepsilon_{p,i}$ in the example, the remaining energy

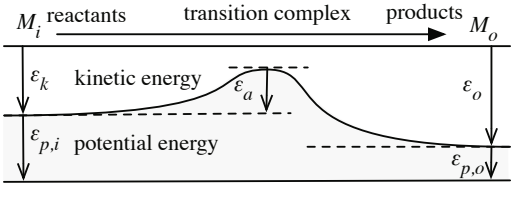


Fig. 1. Energy diagram of an exothermic reaction. The total energy is conserved.

is released, returning to the pool E_k, and the reaction is exothermic. Conversely, the corresponding reverse reaction (from right to left) is endothermic, absorbing energy.

Parameters such as the mapping from molecule to potential energy, as well as the mapping from reaction to activation energy, are left open to be instantiated by the designer to a specific artificial chemistry. Section 3 will show a simple instance of the framework, able to solve the string elongation problem.

2.1 Related Work

Related energy models aim mostly at the study of real chemistry or biology, especially the origin of life. These existing models do not seem directly appropriate in an algorithmic chemistry context. Some are too complex (down to the quantum level [8]), others too simplified or too specific (e.g. focusing on equilibrium states [9], or on catalytic reactions [10]). Most are unable to handle constructive systems: Many require the designer to specify rate and energy parameters for each species and reaction exhaustively [11]; or assign kinetic coefficients at random [12,13], without considering the reactants' composition or shape. Some handle shape explicitly [9,14] but not activation energy.

3 Elongation Control in an Algorithmic Chemistry

A simple string rewriting chemistry is used to illustrate the role of energy in length control. It is a subset of [15] selected on purpose to show a very aggressive elongation behavior. The chemistry consists of polymers strings $s = \Sigma^*$ of arbitrary length over an alphabet of 4 symbols $\Sigma = \{A, M, s, n\}$. The first symbol of a string implicitly defines the string rewriting operation applied to this molecule as follows

$$\begin{aligned} \mathbf{A}\Psi + \mathbf{M}\Omega &\longrightarrow \Psi\Omega && \text{(join)} \\ \mathbf{s}\alpha\beta\Omega &\longrightarrow \alpha\Omega + \beta\Omega && \text{(split)} \\ \mathbf{n}\Omega &\longrightarrow \Omega && \text{(neutral)} \end{aligned}$$

where α, β are arbitrary symbols and Ψ, Ω are strings. Strings starting with A or M are in normal form while strings starting with s or n are transient molecules that immediately undergo as many reduction steps as necessary to reach their normal form. For example, the following two rewriting steps are considered one reaction step (the intermediate product $sAMsAM$ is immediately reduced):

$$AsAM + MsAM \longrightarrow (sAMsAM \longrightarrow) AsAM + MsAM$$

Strings in this chemistry tend to increase in length due to the join reaction. Moreover, contrary to nature, mass (in number of symbols) is not conserved, and the split operation nearly doubles the mass of the input.

3.1 Evolution Experiments

The initial population consists of $C = 100$ identical reaction vessels (or "cells") containing a manually-designed molecular organization that catalyzes its own production: $\{AsssAM, MsssAM\}$. In the absence of resource constraints its concentration rises exponentially.

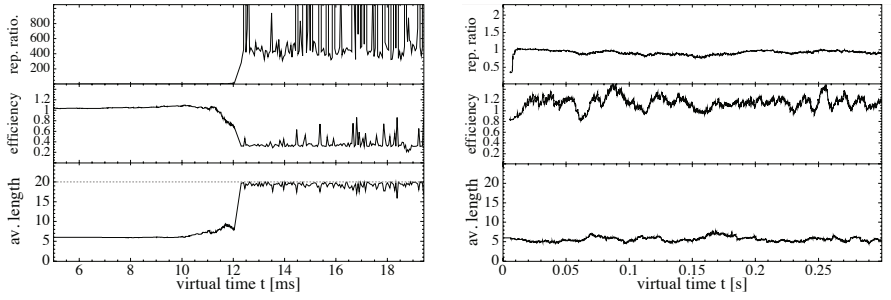
Using a cell growth-division metaphor, cells divide whenever the number of "membrane" molecules (here: molecules starting with symbol M) reaches a threshold of $N = 1000$, in which case the following operations are carried out: (i) A new offspring cell is generated, (ii) half of the molecules of the dividing cell are randomly selected to move to the offspring, (iii) one of the migrated molecules is randomly mutated, and finally, (iv) the new cell displaces one of the already existing cells, maintaining a constant cell population. Natural selection arises because higher production of membrane molecules leads to earlier division. Thus the most efficient autocatalytic set is expected to survive.

We simulate three scenarios, shown below: one with no restrictions, one with a fixed length threshold, and one with energy control.

Unbounded Growth of Unconstrained System: A significant fraction of all mutations lead to a set of molecules with *infinite closure* [4] that has the potential to reach an infinite sequence space when reacting among each other. Consequently, without any length restriction, such a mutation may easily result in the accumulation of ever-growing strings. The resulting exponentially rising CPU and memory requirements prevent the simulation of such systems to be carried out on today's computers. We performed 20 simulations, none of which survived 10 generations without exhausting our machine's resources.

Low Efficiency Under Arbitrary Length Thresholds: We tried two simple methods to prevent strings from growing infinitely: to destroy molecules longer than a certain threshold $l = 10, 20, 30, 100$ (arbitrarily chosen); or to truncate molecules longer than l. Both methods prevent elongation but have an undesired side effect: After several generations, the number of A and M symbols becomes unbalanced due to the stochastic distribution of molecules during cell division. The initial selective advantage of producing more Ms than As goes along with ceasing the production of As necessary for sustained replication. On the other hand, some cells first create additional As while still producing Ms. Consequently, the rate of reactions among As and Ms rises due to the law of mass action. Since these reactions still generate Ms, the membrane production rate increases, too. Such mutants quickly take over the population since their reproductive ratio is about 500 times higher compared to the original program (see Fig. 2(a)). However, this high productivity comes along with a lower *efficiency*, measured as the surface to volume ratio, where the surface is the number of membrane molecules and the volume is defined as the total number of symbols in the cell. At the same time the average molecule length almost reaches threshold l, indicating that the cells fully exploit the length restriction. A typical representative for $l = 20$ is the following multiset:

$$\{278\,MssssssssssssssssssssssssssssAM, 241\,MssssssssssssssssssssssssssssAA,$$
$$1974\,AsssssssssssssssssssssssssssssAM, 553\,AsssssssssssssssssssssssssssssAA\}$$



(a) Length truncation at 20 symbols. (b) Energy inflow $\rho_E = 10^7$ units/s.

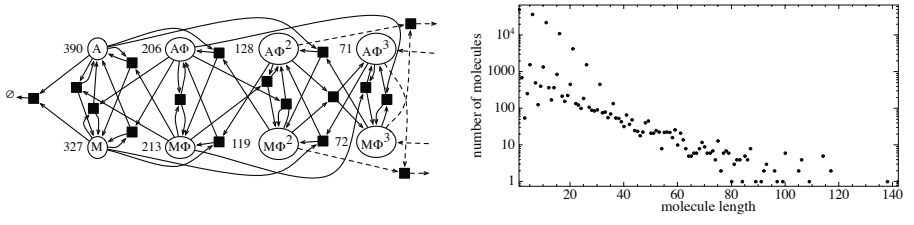
Fig. 2. Simulation results with length restriction (left) and energy control (right). Top to bottom: Average reproductive ratio and average efficiency, both normalized w.r.t. the values of the initial program; and average molecule length.

Emergence of Self-Replicating Clusters Under Energy Control: Instead of externally applying hard length restrictions, the energy framework embeds a notion of resource awareness into the chemistry itself. Less energy slows down reactions and hence the production of new symbols. This method turns out to be very effective in restricting the length of evolved solutions.

We limit the total energy (kinetic plus potential energy) of the system and use a model that, unlike in nature, allows mass to be freely converted to energy and vice-versa: The potential energy of each molecule s is set to its length: $\varepsilon_p(s) = |s|$. The activation energy of each reaction r is set to the difference of potential energy between products and reactants: $\varepsilon_a(r) = \max(0, \varepsilon_p(M_o) - \varepsilon_p(M_i))$, where M_o is the product multiset and M_i is the reactant multiset. Hence, in this chemistry, reactions that build up mass are endothermic, requiring kinetic energy, whereas reactions that destroy symbols are exothermic. Cells receive a constant inflow of energy needed for their growth, whereas the total energy of a displaced cell is lost.

We started with a moderate energy injection rate of $\rho_E = 10^5$ units/s, which results in linear growth. After 100 generations most of the cells still run the initial program; no better solution could be found. Unlike before, now the system cannot increase the reaction rate by excessively producing A molecules. Any production of molecules decreases the temperature and makes endothermic reactions less frequent. Thus a low energy injection rate successfully eliminates mutants with infinite closures.

When increasing energy injection by two orders of magnitude to $\rho_E = 10^7$ units/s, a qualitatively different phenomenon arises: The initial program is able to grow exponentially after each cell division because sufficient energy is available. During cell growth kinetic energy is shared by an exponentially increasing number of molecules. This cools down the cell which gradually returns back to linear growth. Arising mutants that contain syntactically infinite closures start to explore the sequence space by generating longer molecules, but due to the energy restriction the cells are limited in doing so extensively. The existence of longer strings leads to viable mutants that incorporate these new molecules while still being able to survive, i.e. they still have comparable reproduction ratios; the average reproductive ratio remains more or less constant over the whole simulation run (see Fig. 2(b)).



(a) A typical cluster of molecules forming an infinite autocatalytic set.

(b) Histogram of molecule lengths after 100 generations.

Fig. 3. Simulation results with energy control for $\rho_E = 10^7$ units/s

One of the inventions that is often observed in cells after 100 generations is a cluster of molecules of different lengths that all react among each other, as depicted in Fig. 3(a). These molecules are structured as repeated sequences of the duplicate and split pattern $\Phi := sssAM$. In Fig. 3(a), Φ^n represents a string that is the n-fold concatenation of the Φ pattern. The numbers next to the molecules denote their concentration in one of the observed vessels. The cluster contains a lot of very short molecules (A and M), which do not contain the necessary information to replicate themselves. However, they react with a small number of larger molecules that contain multiple copies of the copy and split motif. These longer molecules are maintained at a lower concentration, which exponentially decreases with their length. Even though the cells start to produce larger strings, the average molecule size remains constant and the population maintains its efficiency as shown in Fig. 2(b). Finally, Fig. 3(b) shows the length distribution of a typical simulation run after 100 generations. The cluster consists of molecules of size $5n+1$: $1, 6, 11, \ldots$, visible on the upper left part of Fig. 3(b). Cells containing such clusters are prominently present in the population.

When we further increase the inflow of kinetic energy to $\rho_E = 10^8$ units/s, the cells grow exponentially without energy shortage. Consequently, mutants start to explore the sequence space more aggressively and without hindrance which leads to the same symbolic imbalance as in the fixed length threshold cases.

4 Discussion

Our results indicate that energy control not only helps keeping reasonable resource bounds but also helps to improve the qualitative behavior of the chemistry and steer its evolution. The elongation control experiments show that string length can be kept under control, in spite of an instruction set which is rather aggressive in terms of elongation. The obvious but not optimal solution of a hard length restriction is not needed. Moreover, qualitatively different phenomena are observed according to the amount of energy injected.

A similar elongation phenomenon was described in [3], where a solution based on multi-level selection is also applied. Our method, however, avoids counting on hardware-dependent parameters such as CPU run time.

The experiments reveal a connection between the injected energy and the exploratory capability of a population of cells: If the injected energy flow is too low, the cells grow linearly and the population does not find better solutions. For very high energy injection rates our cells exhibit unbounded growth which leads to symbolic imbalances resulting in a very high reproductive ratio of the affected cells. This is followed by the sudden death of the whole population. Even if the mechanisms behind this behavior are not comparable with those that trigger the elongation catastrophe in [3], interestingly, the resulting effect of the "rise and fall of the fittest" is the same. There exists a range of moderate energy injection rates for which the system is able to survive *and* explore sequence space. "Clusters" of molecules emerge, which altogether form an autocatalytic set. Even though the closure of the set is syntactically infinite, the dynamics of the energy-aware algorithm makes reactions that form long strings more unlikely. This nicely reflects the nature of biochemical reaction system where more and more complex molecules evolve over time in the presence of enough energy.

5 Conclusions

The importance of energy has not been emphasized yet enough, in the context of algorithmic chemistries aimed at performing emergent computation tasks on traditional von Neumann computers. We attempt to fill this gap with an energy framework allowing such algorithmic chemistries to behave thermodynamically and kinetically similar to real chemistry, yet in a simplified form. In spite of "copying" nature's energy model, the framework still allows for exploration of "life-like computations", since the degrees of freedom to parametrize the chemistry still remain very large.

We illustrated the framework's usefulness in a string elongation control task. Even a very aggressive string rewriting chemistry can be restrained by restricting energy alone. Only then and only for a narrow range of energy injection rates the chemistry's state space exploration can be controlled accurately. An interesting topic deserving further investigation is how to derive the optimum energy injection rate automatically.

For the future we also aim at evolving programs where the quality of the computed solution is rewarded with energy. Mass and energy conservation could

be combined in a more realistic setup. Other exploration venues include computations by artificial metabolisms, as well as the evolution of chemical Carnot cycles [16] with their ability to build up information.

Acknowledgments. This work has been supported by the Swiss National Science Foundation and the European Union, through SNF Project Self-Healing Protocols and FET Project BIONETS, respectively. We would also like to thank Stephan Bürgi for reviewing the chemical physics part of our work.

References

1. Fontana, W., Buss, L.W.: "The arrival of the fittest": Toward a theory of biological organization. Bull. Math. Bio. 56(1), 1–64 (1994)
2. Banzhaf, W., Lasarczyk, C.W.G.: Genetic programming of an algorithmic chemistry. In: Genetic Programming Theory and Practice II, pp. 175–190. Springer, Heidelberg (2004)
3. Decraene, J., Mitchell, G., McMullin, B.: Unexpected evolutionary dynamics in a string based artificial chemistry. In: Proc. 11th Int. Conf. Sim. Synthesis of Living Systems (ALife XI), pp. 158–165. MIT Press, Cambridge (2008)
4. Dittrich, P., Speroni di Fenizio, P.: Chemical organization theory. Bull. Math. Bio. 69(4), 1199–1231 (2007)
5. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. 81(25), 2340–2361 (1977)
6. Gibson, M.A., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. A 104(9), 1876–1889 (2000)
7. Upadhyay, S.K.: Chemical Kinetics and Reaction Dynamics. Springer, Heidelberg (2006)
8. Benkö, G., Flamm, C., Stadler, P.F.: Explicit collision simulation of chemical reactions in a graph based artificial chemistry. In: Capcarrère, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., Timmis, J. (eds.) ECAL 2005. LNCS (LNAI), vol. 3630, pp. 725–733. Springer, Heidelberg (2005)
9. Lancet, D., Sadovsky, E., Seidemann, E.: Probability model for molecular recognition in biological receptor repertoires. PNAS 90, 3715–3719 (1993)
10. Pereira, J.: A biochemistry-inspired artificial chemistry: LAC. In: Proc. 12th Portuguese Conf. Art. Intel. (EPIA 2005), pp. 79–84 (2005)
11. Gordon-Smith, C.: Evolution without smart molecules. In: "Extending the Darwinian Framework: New levels of selection and inheritance" Workshop at ECAL 2007, 9th Euro. Conf. ALife (2007)
12. Bagley, R.J., Farmer, J.D.: Spontaneous emergence of a metabolism. In: Proc. 2nd Int. Conf. A Life, pp. 93–140. Addison-Wesley, Reading (1992)
13. Fernando, C., Rowe, J.: Natural selection in chemical evolution. J. Theor. Bio. 247(1), 152–167 (2007)
14. Takeuchi, N., Hogeweg, P.: Evolution of complexity in RNA-like replicator systems. Biology Direct 3 (2008)
15. Tschudin, C.: Fraglets - a metabolistic execution model for communication protocols. In: Proc. 2nd Ann. Symp. Auton. Intel. Net. Sys., AINS (2003)
16. Smith, E.: Thermodynamics of natural selection II: Chemical carnot cycles. J. Theor. Bio. 252(2), 198–212 (2008)

Are Cells Really Operating at the Edge of Chaos?
A Case Study of Two Real-Life Regulatory Networks

Christian Darabos[1,2], Mario Giacobini[2,4], Marco Tomassini[1],
Paolo Provero[2,3], and Ferdinando Di Cunto[2,3]

[1] Information Systems Department, Faculty of Business and Economics
University of Lausanne, Switzerland
[2] Computational Biology Unit, Molecular Biotechnology Center
University of Torino, Italy
[3] Department of Genetics Biology and Biochemistry
University of Torino, Italy
[4] Department of Animal Production Epidemiology and Ecology
University of Torino, Italy

Abstract. It has been suggested that the cells of living organisms are functioning in a near chaotic regime called *critical*, which offers a trade-off between stability and evolvability. Abstract models for regulatory networks such as Kauffman's Random Boolean Networks certainly point in that direction. In this work, we applied the essence of these models to investigate the dynamical behavior of two real-life genetic regulatory networks, deduced in two different organisms. Moreover, a novel, more biologically accurate, way individual genes respond to activation signaling is investigated. We perform numerical simulation and successfully identify contexts in which our model's response can be interpreted as critical, thus most biologically plausible. We also discover that results are comparable in both studied organisms.

1 Introduction

Today, it is widely accepted that genes, considered as static elements of information, are the central pillar of biological evolution, and therefore of life as we know it. On the contrary, much is still unknown about genes as part of a dynamical biological system. Highly complex regulating interactions take place amongst genes to permit the evolution of the organism overtime. These interactions can be represented as genetic regulatory networks (GRNs), showing the influence of a gene on the others. Sadly, interactions within these networks are very subtle, intricate, and ill-understood. Nevertheless, GRNs are the next big thing, and are at the center of tremendous research efforts in the biological community. The quantity and quality of results in the field, thanks to modern high-throughput molecular genetics methods, are bound to follow the same exponential trend as the gene sequencing did in its time. In the meantime, however, it is possible, and useful, to abstract many details of the particular kinetic equations in the cell and focus on the system-level properties of the whole network dynamics.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 281–288, 2011.

A possible dynamical model for GRNs was proposed in the late 60's by Kauffman [1] and is known as Random Boolean Networks (RBNs). This abstraction is very attractive from its simplicity, yet unveils interesting dynamical phenomena about how the network structure and the gene-on-gene interactions are at the center of the resilience to transcriptional errors, and evolvability of GRNs. The dynamics of RBNs can be discriminated in two regimes: the *ordered* regime, where the system tends towards less changes in the gene activations, thus more stability to transient faults, and the *chaotic* regime, where gene activation changes frequently, thus less stability and more evolvability. It has been suggested that biological cells operate at the border between order and chaos, a regime called *critical* or *edge of chaos* [2,3,4]. A way to visualize this phase transition makes use of *Derrida plots* [5], which provide an absolute way of classifying RBN systems according to their dynamical behavior.

In a previous work [6], we have highlighted the weaknesses of Kauffman's original assumptions, that is, the random topology of the networks and the total synchronicity of events. The new model, although more faithful to present knowledge about biological networks, still suffered one identical flaw as the original one, that is, the gene-on-gene interactions are drawn at random. In the present paper, as substrate for RBNs we use two subnetworks of real-life GRNs where the interactions between the genes are known with a good level of confidence. In addition, we take advantage of extra information, namely the actual activating or repressing effect of the genes, to extend the RBN update function proposed by Li *et al.* [7] and Stoll *et al.* [8]. This fills another gap of the original model where the nodes' update function are completely random. We analyze under which conditions real-life GRNs, as a structure for RBNs, can be considered as lying at the *edge of chaos*.

Section 2 will describe RBNs with more details, including their original update functions, the regimes, and the Derrida plots. In Section 3, we describe the embryonic stem (ES) cell and the yeast cell-cycle GRNs that we use as a base for RBNs. Section 4 is devoted to describing the model and the computer simulations in details, results of which will be interpreted and analyzed in Section 5. In the final section we summarize our contribution to the field and outline some path our research may take.

2 Random Boolean Networks and Regimes

Random Boolean Networks (RBNs) have been introduced as a highly simplified model of GRNs. Each node has a connectivity K and represents the Boolean state S_i of a gene. Each directed edge illustrates the influence of a gene on another. Nodes also possess a distinct random function that decides state changes according to the state of all genes that have an incoming edge. The probability p set to the update function with which a gene's state at the next time-step is *active* can be adjusted, but then remains constant during the simulations. The state S of the system is defined as the ensemble of all the nodes states S_i. The state changes are fully deterministic, synchronous and instantaneous. Therefore, these systems, when starting from an arbitrary state S at time $t = 0$, will go

through a set of transient states before eventually cycling in a subset of states called an *attractor*. In our research, we aspire to inject modern knowledge into the original RBN model, making it more biologically plausible.

Firstly, even if their exact values are unknown, it is clear that gene update functions should not be random. Gene expression rests on the combined effect of incoming proteins that can have a *activating* (+) or *repressing* (−) action on their target genes. Stoll *et al.* [8] have proposed a simple additive dynamical rule that characterizes the temporal evolution of the state variable. They consider that both the activating and repressing factors have the same weight, and thus, the state of a target gene at the next time-step $S_i(t+1)$ will be: *active* (1) if it receives a majority of *activating* components from already active genes, *inactive* (0) it receives a majority of *repressing* components, or the state of the target gene will remain unchanged in case the number of *activating* and *repressing* inputs are equal.

Another questionable assumption of the original RBNs model is the totally random interaction among genes with a fixed connectivity K. Thanks to the recent developments in high throughput genomic and biochemical techniques, small parts of real-life GRNs have been discovered with data sufficiently reliable to specify highly probable interaction, with different confidence rates. We have selected the core transcriptional network in embryonic cells published by Chen et al. [9] and a portion of the yeast cell-cycle by Li *et al.* [7] as a substrate for our RBN model. The next section describes these two networks in detail.

Original RBNs go through a phase transition by tuning parameters such as K and probability p of expressing a gene. Considering current knowledge about GRNs, some of Kauffman's properties of the model are now subject to criticism. In Aldana's scale-free model [10], where the output degree distribution follows a power-law $p(k) \sim k^{-\gamma}$, this phase transition is obtained by adjusting γ. In our case, we use real-life networks and not hand-made ones, and thus we cannot tune any property of the network topology to obtain the desired critical regime, or even identify the regime of one of our networks. Instead we use a dynamical property of the whole system, called Derrida plots, proposed by Derrida *et al.* [5], used by Kauffman [11] and widely accepted, as a visual way of discriminating the regime in which RBN-like dynamical systems evolve.

This representation can be seen in Figures 2 and 3, and is meant to illustrate a convergence versus a divergence in state space that can in turn help characterizing the different regimes. It uses the Hamming distance H, defined as the normalized number of positions that are not identical when comparing two (binary) strings. These plots show the average $H(t)$ between any two states s_a and s_b and $H(t+1)$ of their respective consecutive state s'_a and s'_b at the next time-step. Derrida plots of system in the chaotic regime will remain above the main diagonal $H(t) = H(t+1)$ longer, crossing the main diagonal earlier and remaining closer to it as the systems approaches the critical regime. Systems in the critical regime remain on the main diagonal before diverging beneath it. Ordered systems remain under the main diagonal at all times. Other possible ways of determining the actual regime of a system would require either scaling of the system size or analyzing the systems' response to failure, but neither is an option in the present work.

3 Real-Life Regulatory Network Cases

The first real-life regulatory network used in our model, proposed by Chen *et al.* [9], is a part of the transcriptional regulatory network in embryonic stem (ES) cells inferred from ChIP-seq binding assays and from gene-expression changes during differentiation. We added activating informations (+ and − signs) from gene co-expression data and eliminated redundancies from this network. We attributed to each regulatory interaction the sign of the correlation coefficient between the expression profiles of the two genes involved in the time-course experiments on differentiation of ES cells reported in [12]. The second one, as described by Li *et al.* [7] and used by Stoll *et al.* [8], is the yeast cell-cycle. Both networks, depicted in Figure 1, have eleven genes.

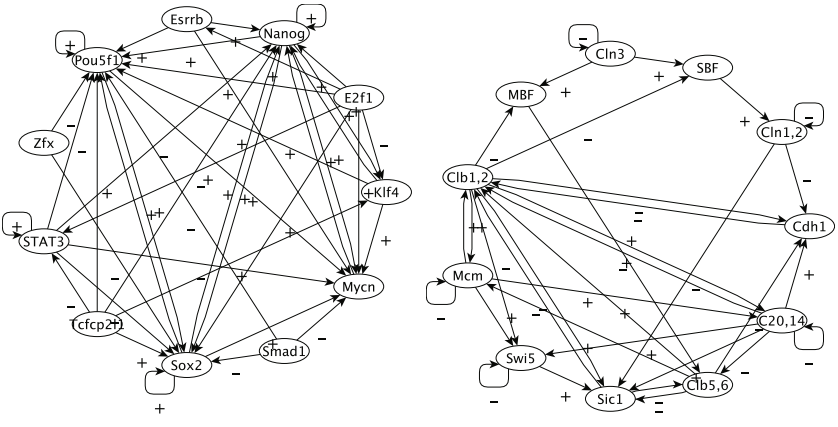


Fig. 1. Genetic Regulatory Networks. A representation of (a) the transcriptional regulatory network in ES cells and (b) the yeast cell-cycle regulatory network. Arrows point from transcription factor to the target gene.

We have statistically studied both networks' degree distributions, concluding that none of the input or output degree distributions show any similarities with either Kauffman's original random topologies, or with Aldana's scale-free Boolean networks. Thus the need to use Derrida plots to determine the regime of our model. In this work, we abstract details of the genes themselves, as their individual properties do not have any consequences on the systems dynamics, beyond their activating or repressing effect.

4 Model and Simulations

In the original RBN model, each node was assigned a deterministic distinct random update function (RUF). Nowadays, we believe that the Boolean function is closer to an additive function where the influence of the genes upstream to the one concerned, along with its own current activity state, could be summed in a

way that takes into account the *activating* or *repressing* effect of each influencing node. Inspired by the work of Stoll *et al.* [8], we propose an update function shared by all genes that takes into account the fact that *activating* and *repressing* components could have unbalanced effects. A gene would require a majority by more than one active gene to switch states, and therefore, we introduced a *threshold* parameter (T) which has to be met in order for a gene to become active. Thus a gene's activation state at the next time-step is now given by:

$$S_i(t+1) = \begin{cases} 1\ (active) & \text{if } \sum_j S_j^+ > T \times (\sum_j S_j^+ + \sum_j S_j^-) \\ 0\ (inactive) & \text{if } \sum_j S_j^+ < T \times (\sum_j S_j^+ + \sum_j S_j^-) \\ S(t) & \text{otherwise} \end{cases}$$

Where S_j^+ is the state of a promoter of the target gene and S_j^- is a repressor gene. Moreover, as some gene of our model might not have any repressors, and, if activated, should not remain in that state permanently, we add a *decay* component. In the case where an active gene has no repressor at all, we switch it to inactive manually at the next time-step. This update function is equivalent to Stoll's in the case where $T = 0.5$. Moreover, it can be easily proven that all rules in this class correspond to a subset of the RUFs. We call our model for update function the Activator Driven Additive function (ADA).

Just the same way the probability p can change Kauffman's original systems' regime from chaotic to ordered for a given connectivity K and set of RUFs, the T parmeter in our model can change its regime. In the following section, we show for which values of T, respectively p, our model of real-life topology-based Boolean Networks using ADA, respectively RUFs, exhibits a phase transition, comparing the dynamics of the two update functions.

The space of all possible states for a given RBNs, i.e. with a single set of RUFs, is 2^N, where N is the number of genes in the system. In our case, $N = 11$, therefore there are $2^{11} = 2048$ possible states. In the case of ADA, where all nodes share the same function, exhaustive enumeration is possible. On the other hand, the set of possible RUFs, even for a reasonably small subset of genes, makes exhaustive enumeration impossible for original RBNs. Therefore, in the case of RUF, we make a statistical sampling using numerical simulation of 100 different sets of RUFs for each value of p. At first, T, respectively p, varies in the interval $[0.1, 0.9]$ by steps of 0.1. After we have identified the values of interest T_i, respectively p_i, we have narrowed the interval to $[T_i - 0.05, T_i + 0.05]$ and $[p_i - 0.5, p_i + 0.05]$ with a finer step of 0.01 to identify as close as possible the values T_c, respectively p_c, that are closest to the critical region.

5 Results and Analysis

In this section, we plot the Derrida curves of each case. Figure 2 which shows Derrida plots with steps of 0.1 and then Figure 3 is a finer version, where we adapted the scale to best show the regions of interest with a step of 0.01. As there is only 2^{11} possible states, and thus $H_{max} = 11$, we computed average Hamming distances over exhaustive enumeration of all states. In other words,

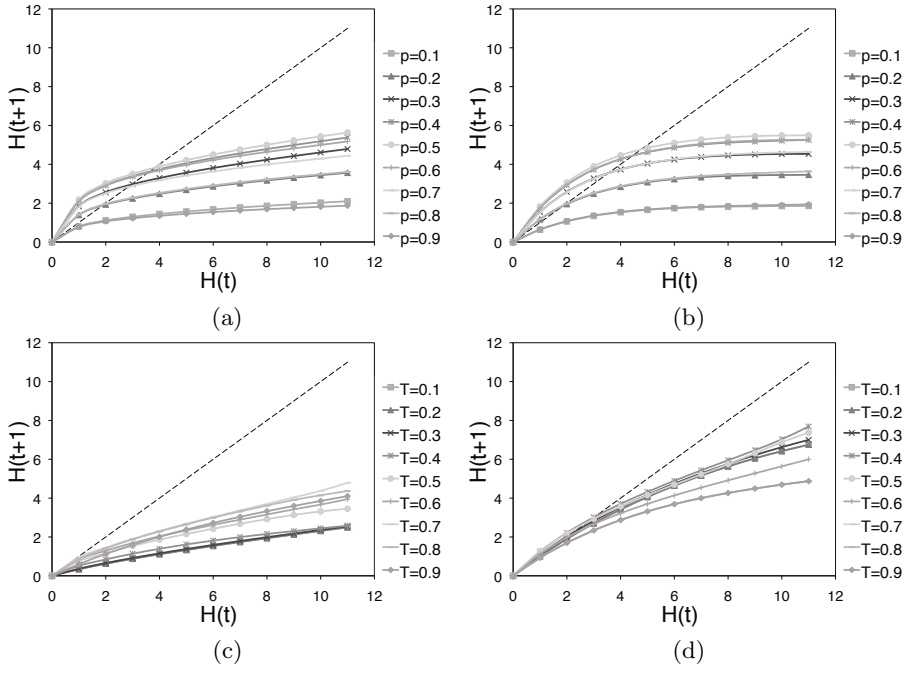


Fig. 2. Derrida plots of Random Update functions (a) and (b), and of Activator Driven Additive, ADA, functions (c) and (d). Derrida plots for real-life networks, (a) and (c) ES cell and (b) and (d) yeast cell-cycle, as substrate for RBN with RUFs. All values of $p \in [0.1, 0.9]$ in (a) and (b), and $T \in [0.1, 0.9]$ in (c) and (d) are shown.

we identified all pairs of states $\{S_a; S_b\}$ that are at a distance $H(S_a, S_b) = 1$ and computed the average Hamming distance of their subsequent states $H(S'_a; S'_b)$, and then moved on to $H = 2$, $H = 3$, ..., $H = 11$. Figure 2 shows both an SE cell and yeast cell-cycles with original model's RUFs and with ADA.

In Figure 2(a)-(b), curves group in pairs according to their values of p, as the RUF functions are symmetrical for values of $p \equiv 1-p$. If not for sampling reasons, pairs of curves would superpose. In the case of transcriptional regulatory network in ES cell, Figure 2(a), the interesting values of $p_i \approx 0.1-0.2$, and symmetrically, $p_i \approx 0.8 - 0.9$. These are the values we chose to investigate with finer steps in Figure 3(a). As for the case of the yeast cell-cycle regulatory network in Figure 2(b), we identified p_i to approximately the same values, developed in Figure 3(b). We have conducted the same analysis on Figure 2(c)-(d). In the case of ADA, there is no symmetry to speak of. The interest regions of T values are $T_i \approx 0.7$ for ES cells. For yeast cell-cycles, we see two regions worth investigating $T_i \approx 0.2-0.3$ and $T_i \approx 0.7$. We show results of finer values of T in Figure 3(c)-(d).

A second set of computer simulations around p_i, the results of which are shown in Figure 3(a)-(b), reveal that in the case of ES cells the critical threshold value is close to $p_c \approx 0.13$, and symmetrically $p_c \approx 0.87$. As for yeast $p_c \approx 0.17$, symmetrically $p_c \approx 0.83$. Finally, the in depth examination of ADA simulation

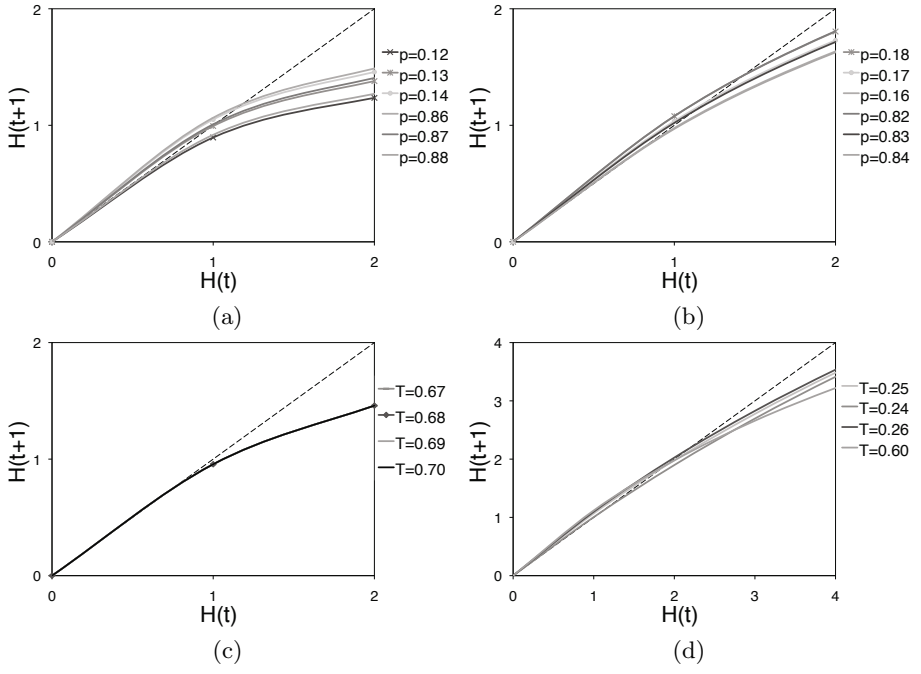


Fig. 3. Derrida plots of Random Update functions (a) and (b), and of Activator Driven Additive, ADA, functions (c) and (d). Derrida plots for real-life networks, (a) and (c) ES cell and (b) and (d) yeast cell-cycle, as substrate for RBN with RUFs. Only values close to the critical gene expression value p_c and T_c are shown.

results for values of T_i demonstrate that results become undistinguishable when the step between T values becomes small. This phenomenon is shown in Figure 3(c), where all the curves of $T = \{0.67, 0.68, 0.69, 0.70\}$ are superposed. This is due to the fact that the ADA function is less sensitive to T for genes with a low input degree. Nevertheless, results remain valid and in the case of ES cells, $T_c \approx 0.68$ and in the case of yeast in Figure 3(d), $T_c \approx 0.25$ or $T_c \approx 0.6$. In this last value of $T_c = 0.6$, several values of very close values of T coincide. From these results, we can conclude that in the present specific cases, both the threshold in ADA and the probability of gene expression in RUF have comparable values in the two GRNs studied in this paper.

6 Conclusions and Future Work

Taking into account recent years' advances in the field of cellular biology, we have proposed to identify under what conditions Kauffman's hypothesis that living organism cells operate in a region bordering order and chaos holds. This property confers to creatures and plants both the stability to resist transcriptional errors and external disruptions, and, at the same time, the flexibility necessary to evolution. We studied two particular cases of genetic regulatory networks found

in literature in terms of complex dynamical systems derived from the original RBN model. In order to do that, we compared the behavior of these systems under the original update function and an novel additive function that we believe is closer to the actual role of living organisms. Results show that there exist values in both update functions that allow the models to operate in the critical region, and that these values are comparable in two different real-life GRNs.

Further investigations of the models at hand will include a full analysis of the attractor space that could offer further evidence of the models' adequacy to real-life situations. In addition, we will extend the current *fully synchronous* timing of state changes with a more biologically plausible one we have proposed in [6], where the topology drives the sequence of individual gene possible activation. Furthermore, we intend to explore the fault tolerance of our models.

Acknowledgements. M. Tomassini and Ch. Darabos gratefully acknowledge financial support by the Swiss National Science Foundation under contract 200021-107419/1.

References

1. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22, 437–467 (1969)
2. Langton, C.G.: Computation at the edge of chaos: Phase transitions and emergent computation. Physica D 42, 12–37 (1990)
3. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality. Physical Review A 38(1), 364–374 (1988)
4. Kauffman, S.A.: At Home in the Universe: The Search for Laws of Self-Organization and Complexity. Oxford University Press, New York (1995)
5. Derrida, B., Pomeau, Y.: Random networks of automata: a simple annealed approximation. Europhysics Letters 1(2), 45–49 (1986)
6. Darabos, C., Giacobini, M., Tomassini, M.: Semi-synchronous activation in scale-free boolean networks. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 976–985. Springer, Heidelberg (2007)
7. Li, F., Long, T., Lu, Y., Ouyang, Q., Tang, C.: The yeast cell-cycle network is robustly designed. Proceedings of the National Academy of Sciences of the United States of America 101(14), 4781–4786 (2004)
8. Stoll, G., Rougemont, J., Naef, F.: Representing perturbed dynamics in biological network models. Phys. Rev. Lett. E 76 (2007)
9. Chen, X., Xu, H., Yuan, P., Fang, F., Huss, M., Vega, V.B., Wong, E., Orlov, Y.L., Zhang, W., Jiang, J., Loh, Y.H., Yeo, H.C., Yeo, Z.X., Narang, V., Govindarajan, K.R., Leong, B., Shahab, A., Ruan, Y., Bourque, G., Sung, W.K., Clarke, N.D., Wei, C.L., Ng, H.H.: Integration of external signaling pathways with the core transcriptional network in embryonic stem cells. Cell 133(6), 1106–1117 (2008)
10. Aldana, M.: Boolean dynamics of networks with scale-free topology. Physica D 185, 45–66 (2003)
11. Kauffman, S.: Understanding genetic regulatory networks. International Journal of Astrobiology 2(02), 131–139 (2003)
12. Sene, K.H., Porter, C.J., Palidwor, G., Perez-Iratxeta, C., Muro, E.M., Campbell, P.A., Rudnicki, M.A., Andrade-Navarro, M.A.: Gene function in early mouse embryonic stem cell differentiation. BMC Genomics 8, 85 (2007)

Cotranslational Protein Folding with L-systems

Gemma B. Danks[1,2], Susan Stepney[2], and Leo S.D. Caves[2]

[1] Computational Biology Unit, Bergen Center for Computational Science, University of Bergen, 5008 Bergen, Norway
[2] York Centre for Complex Systems Analysis, University of York, York, YO10 5YW, United Kingdom

Abstract. A protein molecule adopts a specific 3D structure, necessary for its function in the cell, through a process of folding. Modelling the folding process and predicting the final fold from the unique amino acid sequence remain challenging problems. We have previously described the application of L-systems, parallel rewriting rules, to modelling protein folding using two complementary approaches: a physics-based approach, using calculations of interatomic forces, and a knowledge-based approach, using data from fragments of known protein structures. Here we describe a model combining these two approaches creating an *adaptive* stochastic open L-systems model of protein folding. L-systems were originally developed to model growth and development. Here we also describe extensions of our L-systems models to investigate *cotranslational* protein folding, i.e. folding during protein biosynthesis on the ribosome, which is increasingly thought to play an important role. We demonstrate that cotranslational folding fits very naturally into the L-systems framework.

Keywords: Cotranslational protein folding, L-systems.

1 Introduction

Proteins are crucial for life. They carry out numerous essential molecular functions in the cell. The function of a protein molecule is determined by its specific 3D shape or *conformation*. A newly formed protein in the cell rapidly *folds* to its functional conformation. The thermodynamic hypothesis [1] states that this final fold, the *native state* of a protein, is its lowest free energy state. The native state of a protein, under physiological conditions, is determined solely by its amino acid sequence. Many features of protein folding are now well understood [2]. However, predicting the native state of a protein from its amino acid sequence alone and modelling the folding process at the atomic level on timescales greater than a microsecond are still not computationally feasible [3].

Proteins may be the simplest example of a biological complex system. They exhibit emergent properties at a range of spatial scales including: the partial double bond characteristic of the peptide bond; patterns of hydrogen bonding; *secondary* structure such as helices and sheets; and the compact and hydrophobic nature of the protein core. Each property is the result of interactions at lower spatial scales. These and other emergent properties of proteins allow them to fold

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 289–296, 2011.

to stable conformations with specific biological functions. Protein folding itself can be viewed as an emergent phenomenon that results from underlying local interactions. We use L-systems [4,5,6], parallel rewriting rules, to investigate what *global protein-like* characteristics emerge from modelling protein folding at a *local* level using local interactions represented by local rewriting rules. We have previously described the use of L-systems in modelling protein folding using a physics-based approach [7], using a calculation of local interatomic forces to guide rewriting rules, and a knowledge-based approach [8], using data from fragments of known protein structures. Here we describe an L-systems model that combines these two complementary approaches.

Most computational models of protein folding start with a fully formed protein chain. However, in the cell protein folding may occur during protein synthesis (*cotranslational* protein folding [9]). L-systems provide a natural framework for modelling growth. Here we also describe the extension of our L-systems models for *cotranslational* protein folding, i.e. protein folding during the "growth" of the chain.

Firstly, in section 2, we give a brief overview of how we use L-systems to model proteins. In section 3 we summarise the physics-based L-systems model (see [7] for more detail) and the knowledge-based L-systems model (see [8] for more detail). Section 4 describes the integration of these two approaches into a combined model leading to an *adaptive* stochastic L-systems model. Section 5 describes the extension of our L-systems models to incorporate cotranslational protein folding.

2 Modelling Proteins with L-systems

Lindenmayer systems, or L-systems, were originally developed for the mathematical modelling of plant growth and development [4,5,6]. An L-system consists of a set of parallel rewriting rules, or *productions*, and an initial string called the *axiom*. The productions replace a symbol, or *module*, called the *predecessor*, with a string called the *successor* (e.g. $a \rightarrow ab$ replaces a with ab) repeatedly for a number of specified *derivation steps*.

The axiom in our L-systems models consists of the amino acid sequence of a protein using the single letter amino acid code. We then use context-free productions to rewrite each amino acid letter in the axiom with a string that represents its component atoms and bonds using a bracketed system to capture the 3D structure (Fig. 1) of amino acid specific side chains.

For the folding process, we use context-sensitive productions to rewrite the structural state of each amino acid (captured as parameters in the L-system representation), depending on its local environment. The details depend on the particular model used.

We use deterministic L-systems in our physics-based model to rewrite the conformation of each amino acid depending on local interatomic forces. The interatomic forces are calculated using open L-systems, which allow an L-system to communicate with an environmental model.

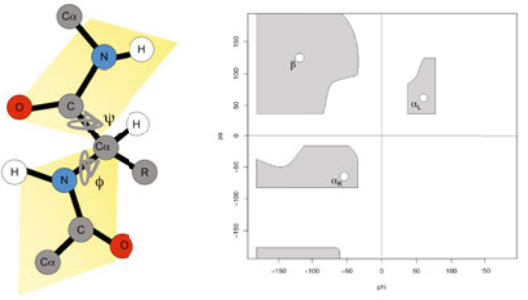


Fig. 1. Local conformations of a polypeptide. Left: an amino acid residue within a polypeptide (a chain of amino acids) consists of an -NH-CαHR-CO- backbone structure, where R represents an amino acid specific side chain (there are 20 different amino acids). Amino acids are joined together by semi-rigid peptide bonds (C-N) causing the four surrounding atoms (CαC-N-Cα) to lie in the same plane (shaded regions). The two main variables in protein conformation are therefore the two back bone torsion angles ϕ (rotation around the N-Cα bond) and ψ (rotation around the Cα-C bond). Right: sterically allowed regions of ϕ/ψ space [10] are shaded in grey on a schematic of a typical Ramachandran plot. These correspond to the most common extended conformations in native structures - the α-helix and β-sheet.

We use stochastic L-systems in our knowledge-based model to rewrite the secondary structure states of each amino acid *residue* with probabilities derived from data on known structures of protein fragments. We use an open L-system environment to store these context-sensitive probabilities.

3 Previous Results

In our physics-based model [7] we define the conformation of a protein using the two backbone torsion angles, ϕ and ψ, for each amino acid residue in the chain (see Fig. 1). Productions alter the ϕ and ψ values of each residue depending on local interatomic forces calculated in an environmental program. This alters the *local* conformation of each residue in parallel, resulting in a *global* change in conformation at each step (see [7] for more detail). The physics-based approach led to the emergence of *protein-like* compact global conformations.

In our knowledge-based model we use data on known protein structures in stochastic rewriting rules. We calculated the frequency of each amino acid type occurring in each of seven secondary structure states used by the DSSP program [11], given the amino acid type and secondary structure state of one amino acid residue either side. We use these frequencies as probabilities, in stochastic productions, of each residue changing its secondary structure state depending on the states and types of its immediate neighbours (see [8] for more detail).

The knowledge-based approach led to the emergence of bands of secondary structure indicating preferred local conformations for certain residues. The proportion of α-helices and β-sheets emerging in the model for different protein

sequences also corresponded well with the structural class of each protein. However, the structures emerging were not necessarily compact, in contrast to the physics-based model, and there was no convergence to a preferred global conformation. This is inevitable when using static probabilities - there is no criterion for choosing one likely state over another.

4 Combined Model Using Adaptive Stochastic L-systems

We have developed a model that combines physics-based and knowledge-based information in order to overcome the problems caused by using static probabilities described above. The local physics that informs changes in backbone torsion angles in the physics-based model is used instead to dynamically alter the probabilities of changing to another secondary structure state in the knowledge-based model (see Fig. 2 for an outline of the combined L-systems model).

Interatomic forces (from an empirical potential) are calculated between each atom attached to the backbone and any other nearby atoms. Changes in both ϕ and ψ are calculated for each residue according to these local forces. The environment also calculates, using typical torsion angle values for each residue in each secondary structure state, the change in ϕ and change in ψ that would be required for each residue to move from its current secondary structure state

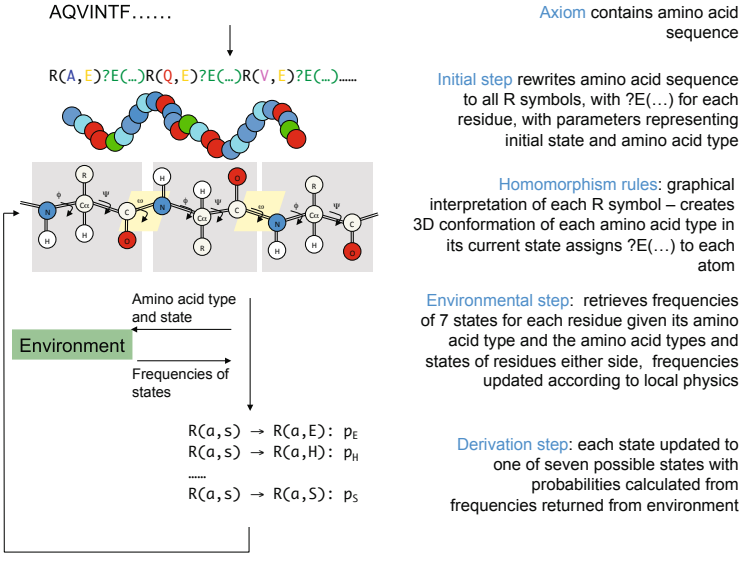


Fig. 2. An outline of the stages in the combined L-systems model. Information at both residue-level (specified in the string) and atomic-level (via homomorphism rules) is sent to the environment via communication modules. The environment retrieves the frequencies for the probabilistic rewriting rules using the information in the residue-level communication modules. The frequencies are altered using physics-based information calculated using the information in the atomic-level communication modules.

to each of the other possible states. These are compared to the changes in ϕ and ψ that were calculated from the local forces. The frequencies are then updated in proportion to these differences. This is repeated at each derivation step - frequency values are updated by scaling values from the previous step according to the forces that result from the new conformation at the current step. This allows the gradual accumulation of a physics-bias into the frequencies, some of which may decrease to zero.

This model leads to a better *protein-like* convergence to a preferred global conformation than the knowledge-based model, while retaining bands of local secondary structure preferences with proportions of α-helix and β-sheet that fit well with the structural class of each protein sequence. Convergence to a preferred global conformation is better in the all-α and all-β structural classes. However, these preferred conformations are not necessarily compact. The final conformation is sensitive to the choice of initial states (particularly in α/β or $\alpha+\beta$ structural classes). An all-extended initial conformation leads to a greater number of residues adopting extended states. Similarly an all-α initial conformation leads to a greater number of residues adopting α-helix states. This may be important in the context of cotranslational folding.

5 Modelling Cotranslational Protein Folding with L-systems

The specific amino acid sequence of a protein molecule is formed during its biosynthesis. Protein-coding genes are *transcribed* into messenger RNA (mRNA) molecules that are *translated* by ribosomes, the macromolecular protein factories of the cell. During translation a ribosome concatenates amino acid building blocks in the order specified by the mRNA being read. Amino acids are concatenated one at a time, by peptide bonds, to form a growing polypeptide which is gradually extruded through a tunnel in the ribosome [12]. Upon exiting the ribosome the polypeptide is free to begin the folding process. Since formation of secondary structure and compact states is faster than protein synthesis [13,14], this simultaneous growth and folding may be important in finding the native state. Furthermore the ribosome itself imposes physical constraints to the initial conformation [12]. L-systems were originally developed to model plant growth and development [4,5,6]. We have extended our L-systems models described above to model the growth of a polypeptide chain and its simultaneous folding, i.e. cotranslational protein folding.

Three main features of cotranslational folding have been simplified and incorporated into our L-systems model: protein synthesis; passage through the ribosome; and extrusion from the ribosome.

Modelling protein synthesis. The full protein sequence is contained in the axiom. A parameter is added to each amino acid module in the axiom to represent its position in the sequence (in the N-terminal to C-terminal direction). A condition is added to the rewriting rules that generate the structural representation of each amino acid. This condition allows one amino acid module

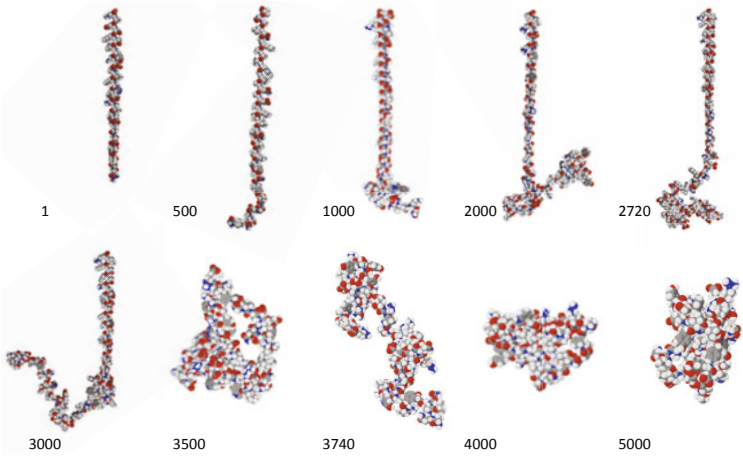


Fig. 3. Images showing protein folding in the cotranslational physics-based model, at derivation step numbers as shown, using the protein sequence barnase (PDB ID: 1bnr). Step 1 shows the initial r residues (here 30) in an initial conformation (here a β-strand) modelling the partially formed polypeptide that is unable to fold inside the ribosome. One residue is added to the C-terminal end every e steps (here 34), while one residue at a time is allowed to start folding at the N-terminal end.

to be rewritten at the C-terminal end of a partially formed chain of length, r (representing the number of residues that are in the polypeptide exit tunnel but are unable to fold), every e derivation steps (representing the rate of protein synthesis): $N > ((n * e) - r)$, where N is the current derivation step number and n is the amino acid number in the sequence.

Modelling restrictions of the ribosome. The polypeptide exit tunnel is approximately 80Å in length [15] and experimental evidence shows that it contains 30-40 amino acid residues [16,17]. We took the lower of these estimates for the number of residues held in the ribosome, r, in our model.

Modelling folding on extrusion from the ribosome. A condition is incorporated into the rewriting rules that alter the secondary structure states of residues (or the backbone torsion angles in the physics-based model) so that only residues at the N-terminal end that are outside of the polypeptide exit tunnel can start folding: $N > (n * e)$. This allows one residue at the N-terminal end to start folding every e derivation steps, as one amino acid structure is added to the C-terminal end. Once the protein is fully formed and all residues are out of the ribosome, all residues can fold in parallel until a specified derivation length.

Results. Protein folding in the cell may be cotranslational if the partially formed polypeptide can adopt a stable conformation. In our physics-based cotranslational model the partially formed polypeptide may rapidly adopt a compact conformation once outside the ribosome (Fig. 3).

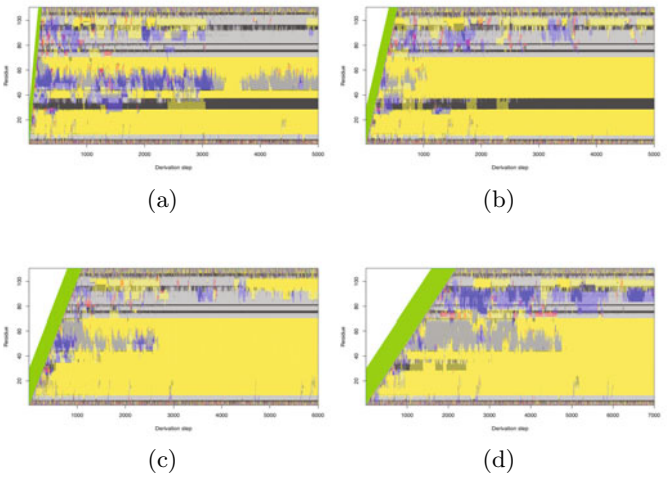


Fig. 4. Changing the rate of growth, e, of the polypeptide chain, using the protein sequence '1exg' in an all-π-helix initial state, in the cotranslational combined model. Plots show the secondary structure states of each residue (y-axis) at each derivation step (x-axis). Yellow = extended, blue = α-helix, red = 3/10 helix, green = π-helix, purple = isolated beta bridge, grey = turn and black = bend. (a) $e = 2$ steps per residue (b) $e = 5$, (c) $e = 10$ and (d) $e = 20$.

The emergence of the final fold through global conformational changes may be dependent on the history of local conformational changes. The rate of protein synthesis may affect this history. In the cell, pauses in translation and the use of rare mRNA codons cause the rate of protein synthesis to vary. Using the combined model (the integrated physics-based and knowledge-based model) we find that cotranslational folding alters the local secondary structure preferences in certain residues and that this is dependent on the growth rate, e, of the polypeptide chain (Fig. 4).

Our cotranslational L-systems models put protein folding into a more biological context. Folding on the ribosome during protein synthesis may be important to finding the native state [14,9]. We have shown that L-systems provide a natural modelling framework for investigating cotranslational protein folding. The L-systems framework facilitates the integration of the growth process of protein synthesis and the developmental process of protein folding, through local conformational changes, into a single model.

6 Summary

Our previous work describes the development of L-systems models of protein folding using two complementary approaches: a knowledge-based approach and a physics-based approach. Here we describe how these models were integrated to produce a combined *adaptive* stochastic open L-systems model, which gives

a greater degree of *protein-like* convergence to a preferred global conformation. We also present a framework for investigating *cotranslational* protein folding using L-systems. This puts protein folding into a more biological context. We demonstrate that L-systems provide a natural framework for modelling the simultaneous growth and folding of a polypeptide chain. Initial results show that the rate of protein synthesis influences the preferred secondary structure preference of some residues in our combined model. Further work will include a more explicit model of the ribosome and will investigate the effects of the rate of protein synthesis across a wide range of protein sequences.

Acknowledgments. This work was funded by the BBSRC.

References

1. Anfinsen, C.B.: Principles that govern the folding of protein chains. Science 181(96), 223–230 (1973)
2. Dill, K.A., Ozkan, S.B., Shell, M.S., Weikl, T.R.: The protein folding problem. Annu. Rev. Biophys. 37, 289–316 (2008)
3. Daggett, V.: Protein folding-simulation. Chem. Rev. 106(5), 1898–1916 (2006)
4. Lindenmayer, A.: Mathematical models for cellular interactions in development. I. Filaments with one-sided inputs. J. Theor. Biol. 18(3), 280–299 (1968)
5. Lindenmayer, A.: Mathematical models for cellular interactions in development. II. Simple and branching filaments with two-sided inputs. J. Theor. Biol. 18(3), 300–315 (1968)
6. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer, Heidelberg (1990)
7. Danks, G.B., Stepney, S., Caves, L.S.D.: Folding protein-like structures with open L-systems. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 1100–1109. Springer, Heidelberg (2007)
8. Danks, G.B., Stepney, S., Caves, L.S.D.: Protein folding with stochastic L-systems. In: Artificial Life XI, pp. 150–157. MIT Press, Cambridge (2008)
9. Kolb, V.A.: Cotranslational protein folding. Mol. Biol. 35(4), 584–590 (2001)
10. Ramachandran, G.N., Sasisekharan, V.: Conformation of polypeptides and proteins. Adv. Protein Chem. 23, 283–438 (1968)
11. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22(12), 2577–2637 (1983)
12. Nissen, P., Hansen, J., Ban, N., Moore, P.B., Steitz, T.A.: The structural basis of ribosome activity in peptide bond synthesis. Science 289(5481), 920–930 (2000)
13. Kubelka, J., Hofrichter, J., Eaton, W.A.: The protein folding 'speed limit'. Curr. Opin. Struc. Biol. 14(1), 76–88 (2004)
14. Basharov, M.A.: Protein folding. J. Cell. Mol. Med. 7(3), 223–237 (2003)
15. Voss, N.R., Gerstein, M., Steitz, T.A., Moore, P.B.: The geometry of the ribosomal polypeptide exit tunnel. J. Mol. Biol. 360(4), 893–906 (2006)
16. Malkin, L.I., Rich, A.: Partial resistance of nascent polypeptide chains to proteolytic digestion due to ribosomal shielding. J. Mol. Biol. 26(2), 329–346 (1967)
17. Blobel, G., Sabatini, D.D.: Controlled proteolysis of nascent polypeptides in rat liver cell fractions. I. Locations of polypeptides within ribosomes. J. Cell Biol. 45(1), 130–145 (1970)

Molecular Microprograms

Simon Hickinbotham[1], Edward Clark[1], Susan Stepney[1], Tim Clarke[2], Adam Nellis[1], Mungo Pay[2], and Peter Young[3]

[1] Departments of Computer Science, [2] Electronics, [3] Biology, University of York, UK

Abstract. Bacteria offer an evolutionary model in which rich interactions between phenotype and genotype lead to compact genomes with efficient metabolic pathways. We seek an analogous computational process that supports a rich artificial heredity. These systems can be simulated by stochastic chemistry models, but there is currently no scope for open-ended evolution of the molecular species that make up the models. Instruction-set based Artifical Life has appropriate evolutionary properties, but the individual is represented as a single executing sequence with little additional physiology. We describe a novel combination of stochastic chemistries and evolvable molecule microprograms that gives a rich evolutionary framework. A single organism is represented by a set of exectuing sequences. Key to this approach is the use of inexact sequence matching for binding between individual molecules and for branching of molecular microprograms. We illustrate the approach by implementation of two steady-state replicase RNA analogues that demonstrate "invasion when rare".

1 Introduction

One sees elegant, evolved design solutions in all levels of life on earth, yet our artificial models of evolution seem limited by comparison. When engaging with the artificial life (ALife) community, it is easy to get the impression that we haven't got the model of genetic algorithms (GAs) quite right yet, and that this is why they are typically only used as optimisers. There is a need for debate about whether the way we think about good engineering is compatible with evolutionary processes [1]. Our project[1] looks at rich ways of building genotype-phenotype (geno-pheno) interactions, as this is how bacteria are known to rapidly adapt to new environments. We are in the early stages of building a bacterial GA. The emulation of biological geno-pheno coupling requires similar coupling between GA and ALife. Other ALife works use GAs [2], but the ALife side tends to spring complete from the GA template, which is never referred to again. We are developing a leakier approach, where the genetic template is an interactive part of the phenotype, with the goal of developing richer ALife behaviours.

We have previously designed a computational metabolome [3], implemented as a network of reactions between a maximum of two molecular agents; it demonstrates that even when expressed with simple computational restrictions, it is straightforward to design a gene regulatory network. To make these networks evolvable, we need to be able to subtly change the nodes (substrates), edges (reactions), and rates in the network,

[1] This work is part of the Plazzmid project, funded by EPSRC grant EP/F031033/1.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 297–304, 2011.

via changes in the binding, reaction and decay of the molecular structures. We need to match the granularity of change in the expressed proteins with the characteristics of mutation that the system possesses. Thus we find that our ALife-GA coupling *demands* an artificial chemistry (AC) as the basis for the geno-pheno interaction. There are three broad classes of AC: abstract (molecular properties specified directly) [4]; shape-based [1]; program-based [5,6,7]. Here we explore the program-based approach, with reference to a set of biological principles that we believe characterise biological systems.

There is a rich history of individual-as-program ALife. Key ones are Avida [5], Tierra [6] and "BF" [7]. All these place heavy emphasis on the genotype: each "organism" has a template code, plus some registers and some energy, which together confer fitness on the individual. The phenotype is essentially a sequence of instructions, plus a miniature processing "factory". In these systems the factory of each organism is not subject to heritable change. We argue that the ALife organism needs to be richer, and use executing sequences as the basis for the chemistry of the organism. By stating that an organism is composed of a *set* of simple executing sub-programs, we can make simpler computational sub-units, draw a closer analogy with functioning proteins, and make more of the processing machinery available to evolve.

The term *microprogram* describes assemblies of the lowest-level instructions (*microcode*) that describes a program at the lowest possible level. By analogy, reactions at the molecule-molecule level form the microcode of the organism [8]. In our representation, molecular microprograms are implemented as sequences of instructions, loosely analogous with the way amino acids fold to form proteins. The "program" emerges via the mixing and reacting of these molecular microprograms. In addition, by basing binding and execution pathways on inexact subsequence alignment [9], we get the benefit of fine-grained switching between execution pathways of the network that forms the high-level program. The novelty here lies in a stochastic chemistry whose molecular species are executing microprograms, and whose binding rates and reactions are emulated via inexact subsequence alignment.

As a demonstration, we have hand-written a molecular microprogram that is capable of copying other molecules that bind to it, including a copy of itself. We call this molecule a *replicase*. We show that it is feasible for mutations to change the efficacy of the microprogram via an "invasion from rare" experimental evaluation.

2 Domain Model of Bacterial Evolution

We describe an abstract model of bacterial evolution in [10], with concepts of the accessory genome and the arrangement of plasmids and chromosomes. The emphasis there is on abstracting genome organisation, but the mechanisms for achieving and maintaining an organised genome are not detailed. Here we define a domain model ([11], as referenced in [12]) of the bacterial system we wish to emulate, and briefly discuss a microprogram-based instantiation of the resulting model.

A naive view of the genome is that it acts as an "workshop manual" for the biological entity. In bacterial systems, the genomic manual is being continually rewritten, from minor typos to new chapters to rearrangement of volumes. What is missing from most evolutionary models is the rich *machinery* for this continual re-editing process.

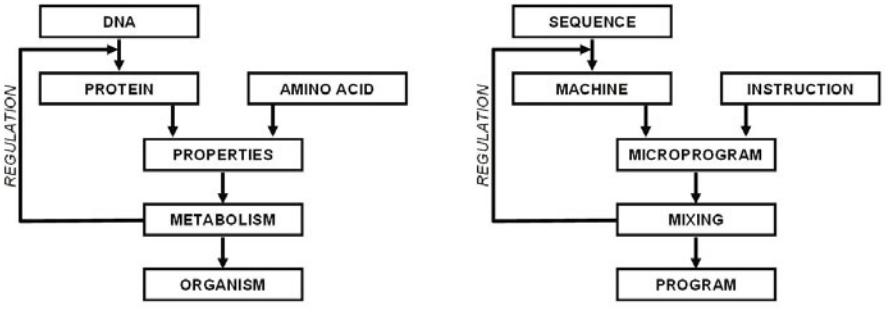


Fig. 1. Domain models of sequence-based executing machines. (a) left: a bacterial metabolism; (b) right: a computer-based simulation.

The copying machinery is specified on the genome that it maintains, and is built the same way as other non-genomic entities in the cell.

Biological genome maintenance is carried out by proteins (folded sequences of amino acids), which are built following a specification on the genome. The functional characteristics of the protein arise from its folded shape, which depends on the chemical properties of the amino acids. As shown in figure 1a, the functionality of the protein is a product of the DNA-based specification of the amino-acid sequence, the physical and chemical properties of the amino acids, and the physical and chemical properties of the protein structure itself. There is a high degree of interdependency between these factors. Proteins' folded shapes are highly structured and often flexible. Reactions occurring between proteins and other molecular entities cause changes in shape and intermolecular forces, which in turn change their reactivity. Over evolutionary timescales, the genetic code develops a deep interaction with these properties. If we wish to emulate these properties, it is imperative to get a granularity of structure that is most appropriate to the goal of open-ended heredity. Figure 1b indicates the components of a computational emulation of the biological system just described. Here a combination of the properties of low-level instructions and the sequnce they are assembled into forms the basis of a microprogram that emulates protein function. The mixing of these proteins in a stochastic chemistry allows the overall program of the system to emerge.

3 Detailed Model

Direct imitation of biology is not currently feasible for all but the simplest systems, since computational overheads are high and biological systems are rarely understood completely. The immediate challenge is to have a specification of a phenotype that allows a rich set of properties to arise such that some template pattern can be used to build a machine. We put rich functionality into the microcodes, and use genetics to exploit their properties. Each microcode is represented by a symbol, and sequences of symbols form the microprogram of a molecule. Each molecule has a set of four pointers, which are manipulated by the microcode to carry out the function of the molecule. Microprogram execution is initiated when one molecule binds to another. When a bind occurs,

Table 1. Properties of an evolving artificial chemistry. General principles (left column). Properties in the molecular microprogram instantiation (right column).

General Properties	Molecular Microprogram Instantiation
single, consistent, molecular representation: for any reaction A+B→C, it must be possible to describe C in the same terms as A and B	molecular microprogram: sequence of microcodes, plus single instance of four pointers
no global controller	contains information only about itself; bioentity little more than a propensity equation
structure of entity forms binding template and function	sequence of microcodes specifies template and execution
binding success proportional to some match function	Smith-Waterman alignment score as basis of bind strength
has a lifetime, eg. defined by a decay rate	decay rate is a function of sequence length
constructed as part of a sequence-reading process	undefined here; straightforward to implement
actions always local to the bound objects	molecular microprograms refer only to themselves and/or their bound partners
actions have cost, and may yield a dividend	execution of each instruction and each attempted bind costs one energy unit; dividend unimplemented: organism requires steady energy influx
actions are not absolute, but relative to the qualities of the objects they refer to	the **=** instruction inserts symbols only at the write pointer **W**: it does not overwrite anything

the two molecules negotiate which one should act as executing program, and which should act as data. The executing molecule's pointers can point to the data molecule.

Our design is motivated by properties of biological systems that we believe to be important. Table 1 lists these general properties, and gives corresponding features of our system. For our initial investigations, we have made assumptions about what the most efficient implementation of a molecular microprogram might be. In this section, we outline the approach we are using to explore the proposals described above.

Our molecular microprograms are analogous to proteins, where sequences of amino acids fold to form molecular machines. We replace the concept of folding with program control flow for a sequence of program symbols. Our molecules have no shape or explicit dimension. The uniqueness of a molecule is encoded in its sequence of symbols. We place heavier emphasis on the concept of binding than do Avida and variants. In functioning microprograms, parts of the sequence describes one or more binding region and when bound, a microprogram is executed from the sequence of instructions beginning at the start of the bind. Thus a binding event triggers a reaction sequence, which is encoded on the molecule and run as a program.

A molecule consists of a sequence of symbols, and four pointers to positions on the sequence: the instruction pointer **I**; the read pointer **R**; the write pointer **W**; and the flow pointer **F**, which is commonly used to reset the position of **I** during iterative execution. Apart from the four pointers, a molecule has no stacks or registers, or access to any global controllers. The symbols in the molecular sequence are instructions that

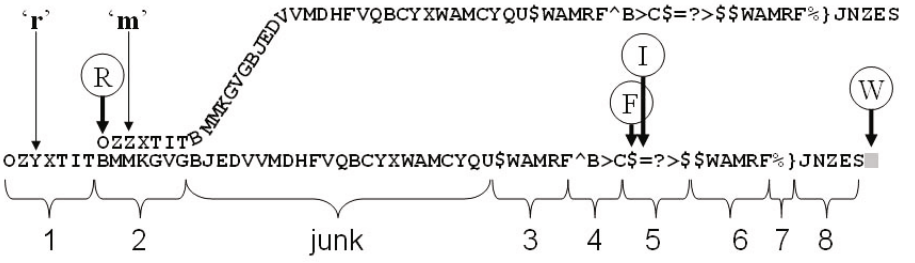


Fig. 2. Composition of a replicase enzyme microprogram. The substrate sequence **m**(top) binds to the executing sequence **r** (bottom). There are eight distinct active regions of the molecule, plus one region of "junk" symbols. The sequence specifies: the initial molecular binding (regions 1 and 2); initialisation of the Read (R), Write (W) and Flow (F) pointers (regions 3,4 and 8); iterative copying of the substrate molecule (region 5); Repositioning the flow pointer to the end of the excecuting molecule (regions 6 and 8); cleaving off the newly created molecule and termination of the microprogram (region 7). Pointers, indicated as letters in circles, show the state of the microprogram as the first symbol of the template molecule is about to be copied to the end of the executing molecule. The two strings differ by a single mutation (indicated by thin arrows), allowing **m** to bind as a substrate to **r** more strongly than **r** binds to a copy of itself.

manipulate the pointers, thereby implementing the molecule's function. We illustrate a replicase molecule in figure 2, which shows eight distinct regions of the microprogram. Execution of the microprogram commences at the start of the bind and proceeds stepwise through each symbol to the right. The diagram shows the positions of the executing molecule's pointers as the first symbol is about to be copied: **I** indicates that the next instruction is **=** (copy). **F** is set to the beginning of region 5, which executes the iterative copy process. **R** is positioned at the start of the template molecule's sequence. **W** is positioned at the end of the executing molecule's sequence. This is where the new molecule is built.

A detailed description of our microcode implementation can be found in [13]. Table 2 provides a summary of these codes, which manipulate the pointers and control the execution pathways of the molecular microprograms.

We use a simple, abstract model cytoplasm [13]. Once created, each molecule floats unbound in the cytoplasm of the organism, and may enconter any other molecule in the system by a process of stochastic mixing. A molecule may bind with another molecule, in which case its microprogram may be executed. The raw materials for the reaction are microcodes, which are assumed to be present at saturation levels. A limiting amount of energy is available at each time increment. Binding and the execution of a single microcode instruction require a single unit of energy to be available. Bound pairs dissociate on termination of the microprogram. Whilst bound, the molecule pair is said to be undergoing a reaction. This process of binding and dissociation continues until the molecule is either destroyed whilst in a bind, or decays whilst unbound.

Binding is a complementary sequence alignment process. The idea is to obtain a probability of binding based on the alignment. The Smith-Waterman (SW) algorithm [9] is ideal for this, since it was designed to give positive scores for similarities that are unlikely to have arisen by chance alone. The algorithm calculates a matrix of aligment

Table 2. Symbols and actions used in the current implementation of molecular microprograms. For a detailed description see [13].

Code(s)	Name	Description
A to **Z**	**n-op**	inactive template code and instruction modifier
$	**search**	shift ***F** to a matching template
>	**move**	shift pointers to the flow pointer
^	**toggle**	switch pointer to molecular bind partner
?	**if**	conditional single or double increment of instruction pointer
=	**copy**	insert at ***W** a copy of the symbol at ***R**
%	**cleave**	split a sequence and generate a new molecule
}	**end**	terminate execution and break the bond between the molecules

scores, based on a matrix of code substitution costs and a gap-scoring function. When two sequences are compared, the SW algorithm finds the strongest local sequence alginment (LSA) between them. SW calculates the tradeoff between the length and the mismatches in the LSA. A perfect alignment of length 5 will score 5. Each mismatch in a subsequence pairing reduces this score by some amount, and the penalty increases for mismatches that are increasingly unlikely, so an alginment of length 10 but with significant mismatches might score 4. Scores of zero or less indicate that the subsequence pairing is likely to have arisen by chance.

For any two molecules i and j, we use SW to detect the LSA for a sequence pair $\phi_{i,j}$, and use the score and the length to define the binding site and derive a probability of binding, $p(\phi_{i,j}) = (s/l)^l$, where s is the SW score and l is the LSA length. For details of how the mismatch penalties are calculated, see [13].

We also use $p(\phi)$ to control program flow: **$** and **?** use a template alignment, and have different operation if an alignment is not found. This "soft" execution pathway is ideal for evolutionary algorithms, since it allows incremental change in microprogram execution that mimics some of the attractive biological properties listed in table 1.

For computational simplicity, on decay a molecule is instantaneously deleted. The chance of decay is a function of the length of the sequence. This is a crude way of ensuring that things that are expensive to build tend to persist in the metabolome, without having molecule fragments floating around and reacting with other, complete molecules in the system. We use a decay probability of $1/L^2$, where L is the length of the sequence. Note that this is "passive" decay. There is scope to build a "destructase" molecule, whose microprogram would chop up anything that bound to it into smaller molecules. The resulting fragments would then be more likely to decay, since they are shorter.

4 Experiments

Inspired by the RNA world model [14], we have conducted studies of metabolic systems composed of molecular microprograms without reference to a genome. In this framework, our molecular species act as their own templates. A replicating molecule is a good candidate for early trials, since only a single species needs to be defined. Our hand-crafted replicase **r** is shown in figure 2. Note that there are very many species

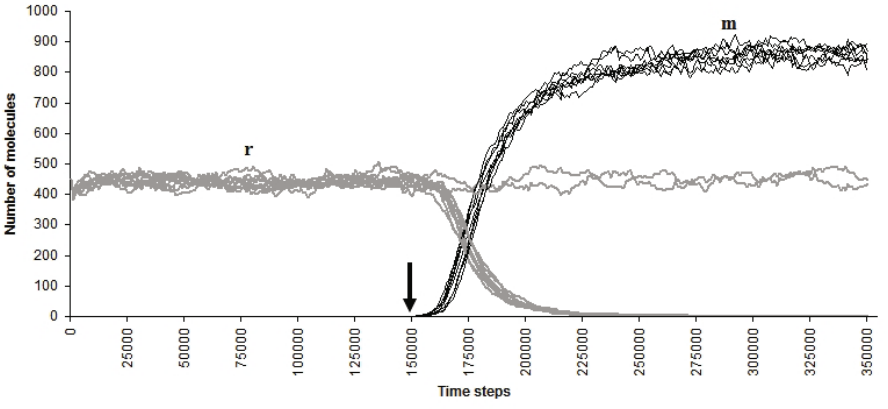


Fig. 3. Invasion when rare. Ten typical runs of a metabolism of weakly binding replicase species $\mathbf{r}$ (thick grey lines) which is invaded (indicated by the arrow) by a single molecule of mutant species $\mathbf{m}$ (thin black lines), which has a stronger binding affinity. Eight of these trial runs demonstrate "invasion when rare" by $\mathbf{m}$. In two cases, $\mathbf{m}$ went extinct, and $\mathbf{r}$ remained at equilibrium.

with replicase functionality in our molecular space. Regions 1 and 2 of the microprogram for $\mathbf{r}$ are complementary sequence alignements of length $l = 7$, score $s = 5.875$ and $p(\phi_{\mathbf{r},\mathbf{r}}) = 0.293$. We have also hand-crafted a variant of $\mathbf{r}$ that could arise by a single mutation. This variant, $\mathbf{m}$, has perfectly matching complementary sites, so $l = 7$, $s = 7$, $p(\phi_{\mathbf{m},\mathbf{m}}) = 1$, and importantly $p(\phi_{\mathbf{r},\mathbf{m}}) = 1$ also: $\mathbf{r}$ binds to $\mathbf{m}$ more readily than $\mathbf{r}$ binds to $\mathbf{r}$.

For experimental trials, we allowed a population of molecule $\mathbf{r}$ to reach equilibrium, then introduced a single molecule of $\mathbf{m}$ at time step 150 000. We ran a metabolism using this specification 100 times. A typical subset of 10 runs is shown in figure 3. The phenomenon of "invasion when rare" is occurring: the competitive advantage of species $\mathbf{m}$ allowed it to replace $\mathbf{r}$ entirely in 88 out of the 100 trials.

5 Conclusion

We have demonstrated that an ALife environment that uses a sequence of instructions as the basis for interaction and program execution can show important biological properties. We intend to use this platform to explore these ideas further, in particular to develop a fully functioning bacterial emulator.

The experiment presented here used a "hand selected" mutation that was designed to give beneficial effects. The experiment demonstrated that a single instance of a beneficial mutation can easily become the dominant species in our system. The idea now is to see how different mutation rates affect a functioning molecular species. Mutation can be implemented straightforwardly by allowing the **`copy`** operator to introduce changes to the copy. It may be possible to determine the "error catastrophe" levels for this system, and use this limit to derive appropriate mutation rates.

Our end goal is to subject this system to evolutionary change. The system is currently an RNA-world model, and could be used to simulate Speigelman's famous Q_β experiments [15]. This would demonstrate that the system is capable of the sort of open-ended heredity that is required for the automated evolution of complex computational machines.

References

1. Bentley, P.J.: Fractal proteins. Genetic Programming and Evolvable Machines, 71–101 (March 2004)
2. Knibbe, C., Fayard, J.M., Beslon, G.: The topology of the protein network influences the dynamics of gene order: From systems biology to a systemic understanding of evolution. Artif. Life 14(1), 149–156 (2008)
3. Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Young, P.: Gene regulation in a particle metabolome. In: CEC 2009, pp. 3024–3031. IEEE Press, Los Alamitos (2009)
4. Clark, E., Hickinbotham, S., Stepney, S., Clarke, T., Young, P.: Encoding evolvable molecules. In: International Workshop on Information Processing in Cells and Tissues (IPCAT), Franscini Ascona, Switzerland, Librix (2009)
5. Pennock, R.T.: Models, simulations, instantiations, and evidence: the case of digital evolution. J. Exp. Theor. Artif. Intell. 19(1), 29–42 (2007)
6. Ray, T., Xu, C.: Measures of evolvability in Tierra. Artificial Life and Robotics 5(4), 211–214 (2001)
7. Bobrik, M., Kvasnicka, V., Pospichal, J.: Artificial chemistry and molecular Darwinian evolution of DNA/RNA-like systems I – typogenetics and chemostat. In: Kelemen, A., Abraham, A., Liang, Y. (eds.) Computational Intelligence in Medical Informatics. SCI, vol. 85, pp. 295–336. Springer, Heidelberg (2008)
8. Danchin, A.: Bacteria as computers making computers. FEMS Microbiology Reviews 33(1), 3–26 (2009)
9. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. 147(1), 195–197 (1981)
10. Stepney, S., Clarke, T., Young, P.: Plazzmid: An evolutionary agent-based architecture inspired by bacteria and bees. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 1151–1160. Springer, Heidelberg (2007)
11. Andrews, P.S., Sampson, A.T., Polack, F., Stepney, S., Timmis., J.: CoSMoS development lifecycle, version 0. Technical Report (in preparation), University of York (2008)
12. Garnett, P., Stepney, S., Leyser, O.: Towards an executable model of auxin transport canalisation. In: Stepney, S., Polack, F., Welch, P. (eds.) CoSMoS, pp. 63–91. Luniver Press (2008)
13. Hickinbotham, S., Clark, E., Nellis, A., Pay, M., Stepney, S., Clarke, T., Young, P.: An abstract metabolism of molecular microprograms. Technical Report (in preparation), University of York (June 2009)
14. Lincoln, T.A., Joyce, G.F.: Self-sustained replication of an RNA enzyme. Science, 1167856+ (January 2009)
15. Mills, D., Peterson, R., Spiegelman, S.: An extracellular darwinian experiment with a self-duplicating nucleic acid molecule. Proceedings of the National Academy of Sciences of the United States of America 58(1), 217–224 (1967)

Identifying Molecular Organic Codes in Reaction Networks

Dennis Görlich and Peter Dittrich

Bio Systems Analysis Group, Jena Centre for Bioinformatics (JCB) and Department of Mathematics and Computer Science, Friedrich-Schiller University Jena, D-07743 Jena, Germany
dennis.goerlich@uni-jena.de, dittrich@minet.uni-jena.de

Abstract. Studying semantics is strongly connected to studying codes that link signs to meanings. Here we suggest a formal method to identify *organic codes* at a molecular level. We define a molecular organic code with respect to a given reaction network as a mapping between two sets of molecular species called signs and meanings, respectively, such that (a) this mapping can be realized by a third set of molecular species, the codemaker and (b) there exists alternative sets of molecular species, i.e., alternative codemakers, implying different mappings between the same two sets of signals and meanings. We discuss theoretical implications of our definition, demonstrate its application on two abstract examples, and show that it is compatible to Barbieri's definition of organic codes. Our approach can be applied to differentiate the semantic capacity of molecular sub-systems found in the living world. We hypothesize that we find an increasing capacity when going from metabolism to protein networks to gene regulatory networks. Finally we hypothesize that during the chemical and Darwinian evolution of life the capacity for molecular organic codes increased by the discovery and incorporation of those reaction systems that contain many molecular organic codes.

1 Introduction

Without doubt information plays a central role in all living systems [1]. The most common approaches that formalize the transmission of information are based on Shannon's information theory [2]. The theory and its derivates have not only been very successfully applied in engineering [3], but also in the life sciences [4,5]. Shannon has explicitly and consciously neglected semantic and pragmatic aspects of information, because they were not required to reach his engineering objectives (cf. Ref. [2], second paragraph). However as pointed out by a growing body of literature [6,7,8] for understanding biological information semantic and pragmatic aspects have to be also considered. Actually there is no formal or computational framework yet that makes semantic aspects as precise as Shannon's theory is.

Our work aims to contribute to this open problem. One approach to analyze semantics in biological systems is to cope with the codes that link signs and

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 305–312, 2011.

meanings. To understand what such a *biological code* is and how it is implemented we base our work on the definition of *organic codes* given by Barbieri [9]. He defines an organic code by identifying the key elements involved in coding. These are *signs*, *meanings*, and *adaptors*. According to Barbieri [10] we can identify following entities to characterize an organic code:

1. a correspondence between two worlds (signs and meanings)
2. a system of molecular adaptors
3. a set of rules that guarantees biological specificity

Adaptors are molecules or molecular complexes realizing an arbitrary mapping between two worlds (signs and meanings) by recognizing elements from both worlds in two *independent* recognition processes. The mapping has been fixed during evolution, but is arbitrary, i.e. another mapping between the same sets could have occurred in evolution.

Examples of organic codes are the genetic code, signal transduction codes [11], sugar codes [12], histone codes (epigenetic codes)[13,14], or the cytoskeleton code [9]. For example, the genetic code is a mapping from the set of codons (signs) to the set of amino acids (meanings). The adaptors are tRNA molecules.

In the remainder of the paper we introduce a method that allows to identify organic codes in reaction networks, demonstrate its properties theoretically and along two abstract example networks, and discuss the potential evolution of molecular organic codes.

2 Identifying Molecular Organic Codes in Reaction Networks

Reaction networks and reaction systems are a general way to represent almost all biological processes and are therefore well suited to be the basis of our definition of organic codes guaranteeing that the definition can be applied to a wide range of systems.

Definition 1 (Reaction Network). *A reaction network* $\langle \mathcal{M}, \mathcal{R} \rangle$ *is defined by a set of molecular species* $\mathcal{M}$ *and a set of reaction rules* $\mathcal{R}$*. A reaction rule* $\rho \in \mathcal{R}$ *is defined by its stoichiometric coefficients* $l_{i,\rho} \geq 0$ *and* $r_{i,\rho} \geq 0$, $i \in \mathcal{M}$*, specifying the left hand side and right hand side of a reaction rule, respectively.*

Let $\mathrm{LHS}(\rho) = \{i \in \mathcal{M} | l_{i,\rho} > 0\}$ denote the set of molecules that appear on the left-hand side of reaction $\rho \in \mathcal{R}$ and $\mathrm{RHS}(\rho) = \{i \in \mathcal{M} | r_{i,\rho} > 0\}$ the molecules on the right-hand side.

We define the algebraic closure of a set of molecular species (cf. [15]):

Definition 2 (Closure). *Given a reaction network* $\langle \mathcal{M}, \mathcal{R} \rangle$*, a set of molecular species* $B \subseteq \mathcal{M}$ *is closed, if all reactions among species in* B *produce only species from* B *(i.e., for all* $\rho \in \mathcal{R}$ *with* $\mathrm{LHS}(\rho) \subseteq B$ *holds* $\mathrm{RHS}(\rho) \subseteq B$*). The closure* $B = G_{CL}(A)$ *of an arbitrary set of species* $A \subseteq \mathcal{M}$ *is defined as the smallest closed set containing* A*.*

Note that this smallest set always exists and is unique [16]. It can be generated by successively adding to the set A those molecules that can be generated by applying the reaction rules of $\mathcal{R}$ to A until no species can be added anymore (which might take an infinite amount of steps).

2.1 Definition: Organic Code with Respect to a Reaction Network

In general, we define a molecular organic code with respect to a given reaction network as a mapping between two subsets of molecular species (called signs and meanings), where (1) the mapping can be realized by a subset of molecular species C of the reaction network and (2) there exists at least a second, different mapping between the same subsets of molecular species that can be realized by a (necessarily different) subset of molecular species C'. Following [10], we call C the codemaker and C' alternative codemaker.

Without loss of generality, we will restrict our following discussion to the most simple but non-trivial codes, which map two signs to two meanings. Additionally we require that this mapping can be inverted. We call such codes *injective binary organic codes* (IBOC):

Definition 3 (Injective Binary Organic Codes, IBOC). *Given a reaction network $\langle \mathcal{M}, \mathcal{R} \rangle$ an injective binary organic code (IBOC) is a function $f : S \rightarrow M$ mapping a two-elementary set $S = \{s_1, s_2\},\ s_1, s_2 \in \mathcal{M}$ to a two-elementary set $M = \{m_1, m_2\},\ m_1, m_2 \in \mathcal{M}$, such that the following conditions hold: There exist two sets of molecular species $C \subseteq \mathcal{M}$ and $C' \subseteq \mathcal{M}$ with*

1. *$s_1 \cup C$ can generate species m_1 but not m_2,*
2. *$s_2 \cup C$ can generate species m_2 but not m_1,*
3. *$s_1 \cup C'$ can generate species m_2 but not m_1, and*
4. *$s_2 \cup C'$ can generate species m_1 but not m_2.*

by application of the reaction rules $\mathcal{R}$. We say that $C \subseteq \mathcal{M}$ can generate species $a \subseteq \mathcal{M}$, if a is contained in the closure of C (i.e., $a \subseteq G_{CL}(C)$). We call S the set of signs, M the set of meanings, C codemaker and C' alternative codemaker.

Note that the alternative codemaker C' implies another code, different from C, on the same sets of signs and meanings. The codes are binary, because we consider S and M as two-elementary sets. Applying the definition of IBOCs always leads to an injective code, because conditions 1 to 4 guarantee that the condition for an injective function f, $x \neq y \Rightarrow f(x) \neq f(y)$, is always fulfilled.

To show that IBOCs formalize the idea of organic codes we go through the properties of an organic code described by Barbieri [10]:

Property 1 (Correspondence between two worlds). As we define two sets, denoted with signs and meanings, an IBOC realizes a correspondence between these sets. Every set represents a world, respectively.

Property 2 (A system of adaptors). The codemaker C corresponds to a system of adaptors, because the molecules of C realize the code by connecting the worlds.

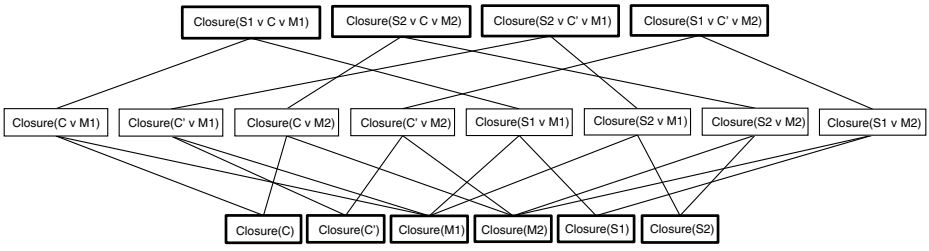


Fig. 1. Closed sets defining an IBOC: The closed sets displayed with bold boxed are necessary for the existence of the IBOC (Lemma 1). The closed sets in the middle row can be present, but need not to be present (e.g., it is possible that $G_{CL}(\{m_1\} \cup C)$ is equal to $G_{CL}(\{m_1\} \cup C \cup \{s_1\})$, example not shown). Note that the smallest closed set and the set containing all elements, both present in any lattice of closed sets, have been omitted.

A single adaptor is a subset of C that is required to map a sign to a meaning. In a given reaction network several codemakers may exist for the same organic code.

Property 3 (A set of rules guaranteeing specificity). Specificity of the rules is guaranteed by the determinism when generating the closed set of a set of molecules and the injectivity of the codes, because they realize a one-to-one mapping.

Lemma 1 (Ten Unique Closed Sets). *Given an IBOC according to Def. 3, the ten closures* $G_{CL}(s_1)$, $G_{CL}(s_2)$, $G_{CL}(m_1)$, $G_{CL}(m_2)$, $G_{CL}(C)$, $G_{CL}(C')$, $G_{CL}(s_1 \cup C) = G_{CL}(s_1 \cup C \cup m_1)$, $G_{CL}(s_2 \cup C) = G_{CL}(s_2 \cup C \cup m_2)$, $G_{CL}(s_1 \cup C') = G_{CL}(s_1 \cup C' \cup m_2)$, *and* $G_{CL}(s_2 \cup C') = G_{CL}(s_2 \cup C' \cup m_1)$ *must be different.*

Proof: follows directly from the definition and from the fact that $G_{CL}(G_{CL}(A) \cup G_{CL}(B)) = G_{CL}(A \cup B)$. □

Note that Lemma 1 implies that S, M, C, C' must be different sets and that, according to Lemma 1, a reaction network must contain at least ten closed sets for an IBOC[1] (cf. Fig. 1).

Given a reaction network we can calculate all IBOCs which are contained in this network by checking all possible combinations of six closures (representing $s_1, s_2, m_1, m_2, C, C'$) against the IBOC-condition (cf. Def. 3 and Fig. 1). This naive but easy implementable approach has a computational complexity of $\mathcal{O}(n^6)$, with n the number of closures of the network.

[1] To be more precise, we would need at least 12 closed sets, because closed sets form a lattice, thus there must be at least two additional closed sets above and below the ten closed sets of the IBOC (Fig. 1).

3 Examples

Figure 2 illustrates the IBOCs contained in two reaction networks. The depicted scenarios define two similar reaction networks. In Scenario A the network consists of eight species $\mathcal{M} = \{A_1, A_2, B_1, B_2, E_1, E_2, E_3, E_4\}$ and four reaction rules $\mathcal{R} = \{A_1 + E_1 \to E_1 + B_1,\ A_2 + E_2 \to E_2 + B_2,\ A_1 + E_3 \to E_3 + B_2,\ A_2 + E_4 \to E_4 + B_1\}$. There are two injective binary organic codes:

$(S = \{\{A_1\}, \{A_2\}\}, M = \{\{B_1\}, \{B_2\}\}, C = \{E_1, E_2\})$ and
$(S = \{\{A_1\}, \{A_2\}\}, M = \{\{B_1\}, \{B_2\}\}, C' = \{E_3, E_4\})$.

In Scenario B the network consists of six species $\mathcal{M} = \{A_1, A_2, B_1, B_2, E_1, E_2\}$ and four reaction rules $\mathcal{R} = \{A_1 + E_1 \to E_1 + B_1,\ A_2 + E_1 \to E_1 + B_2,\ A_1 + E_2 \to$

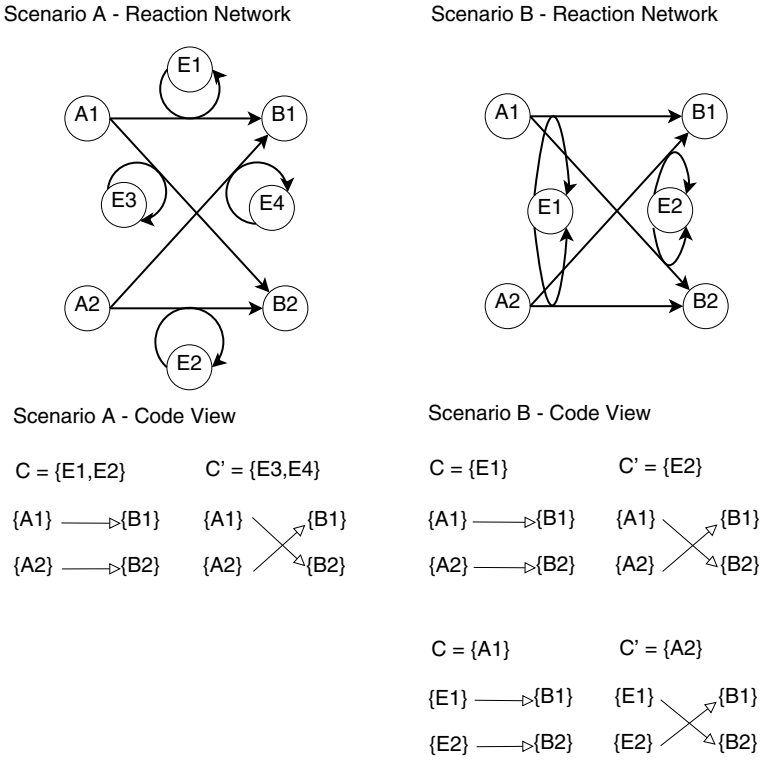


Fig. 2. Illustration of all injective binary organic codes that can be realized by a reaction network: Two scenarios (A, B) are depicted. Scenario A has eight species where four are metabolites and four function as enzymes. Scenario B contains also four metabolites, but only two enzymes. In the top picture the reaction network is depicted. Arrows in the network correspond to the reactions $\mathcal{R}$ that can occur in the network. In the code view the open arrows denote the mapping f between sets of species. This connection may be established by more than one reaction (example not shown).

$E_2 + B_2,\ A_2 + E_2 \rightarrow E_2 + B_1\}$. Here we have four injective binary organic codes:
$(S = \{\{A_1\},\{A_2\}\}, M = \{\{B_1\},\{B_2\}\}, C = \{E_1\})$,
$(S = \{\{A_1\},\{A_2\}\}, M = \{\{B_1\},\{B_2\}\}, C' = \{E_2\})$,
$(S = \{\{E_1\},\{E_2\}\}, M = \{\{B_1\},\{B_2\}\}, C = \{A_1\})$, and
$(S = \{\{E_1\},\{E_2\}\}, M = \{\{B_1\},\{B_2\}\}, C' = \{A_2\})$.

Note that in Scenario A there are only two IBOCs, while in Scenario B four of these exist. In Scenario A it is not possible to take a single enzyme E_i as a sign. For example, $(S = \{\{E_1\},\{E_2\}\}, M = \{\{B_1\},\{B_2\}\}, C = \{A_1, A_2\})$ is *not* an organic code, because there does not exists a second codemaker $C' \subset \mathcal{M}$ that assigns E_1 and E_2 to B_1 and B_2 in a different way than C does. Thus we can say that the appearance of B_1 is a *natural sign* for E_1, because the production of B_1 implies E_1 as a necessary species.

4 Discussion

Evolution of Molecular Codes. Many hypothesis have been made how the genetic code has evolved [17,18,19]. Recently Koonin [19] stated that to understand the evolution of the genetic code we have to understand the evolution of codes in general. The codes defined in this paper may be suitable to understand how codes in general evolve. We hypothesize:

Hypothesis 1: *During the origin of life (chemical evolution) and the evolution of life the number of molecular codes (IBOCs) contained in the reaction systems discovered and incorporated by living systems increased.*

Molecular Codes in Living Systems. If we compare different sub systems of living systems, e.g. metabolism, protein interactions or gene regulation, we suggest that these systems have different capacities for realizing molecular codes. In metabolic networks the mapping between molecules, i.e. reactions, occur only if the linked metabolites are quite common in their biochemical structure. Because of this strong relation less arbitrariness can occur and less codes may evolve. Protein interaction networks might have more molecular codes, as the interactions are not directly connected to the internal structure, but to surface interactions, that may be more fuzzy. In gene regulatory networks the number of codes might be quite high, because the genes might be coupled almost arbitrarily, since there is no structural linkage between them. In summary we hypothesize:

Hypothesis 2: *The molecular systems of cells have different semiotic capacities, i.e. have different number of IBOCs. We propose that metabolic networks, protein interaction networks and gene regulatory networks contains an increasing number of IBOCs, respectively.*

Note that both hypotheses can be tested by applying an IBOC-identifying algorithm on the respective reaction networks.

Relation to Information Theory. Our definition of an IBOC should capture some semantical aspect of biological information. But how is it related to statistical

or information theoretic concepts? A common approach is to equate information with correlation or mutual information between two random sources, e.g. the message and its environment [20]. High mutual information would also be necessary for our IBOCs, but not sufficient. In other words, measuring a correlation or mutual information between two worlds does not necessarily imply that there is a code or a semiotic structure. In addition we need "arbitrariness", represented formally by the alternative codemaker C'. Otherwise we have direct physical causal relationship or a natural sign (cf. [10]).

Generalization of IBOCs. Here, we have restricted our study to a most simple, non-trivial class of molecular organic codes, the IBOCs. This approach generalizes relatively obviously in various directions: (1) We can allow an arbitrary number of signs and meanings. (2) The mapping f needs not to be injective, like as in the genetic code. (3) A sign or meaning could be defined by quantitative properties like concentration ranges or oscillatory frequency ranges. (4) We can consider dynamics by requiring that the mapping between signs and meanings must be realizable by a feasible and specific trajectory. (5) We might consider further algebraic constraints like catalytic closure or self-maintenance [21]. Note that some dynamical properties can already be considered implicitly when dealing with algebraic criteria as we did here (cf. Theorem 1, [21]). E.g. we could require that the union of sign and codemaker generates a chemical organization containing the meaning; otherwise the meaning would appear only as a transient phenomenon. (6) We can measure the arbitrariness of a code by the number of alternative codemakers and how difficult (or likely) it is to make them. (7) With fuzzy-set theory or information theory we can obtain a more contineous organic code concept.

Pragmatics. Here, we have completely neglected the pragmatic aspect of biological information. We have only dealt with a semantic issue: the code, which relates signs to meanings. In the future we have to include pragmatics, by answering questions like: How is the code used? Or, in particular in the light of evolution, how is the code used for self-maintenance and reproduction. Having made the notion of a molecular code precise should ease this work significantly.

5 Conclusion

The definition of a molecular organic code and its specific formal instance the IBOC (Def. 3) allows us to study organic codes and thus semantic structures in reaction systems. It provides a precise instrument to characterize a reaction system to what degree it can realize organic codes and how it may operate as a semiotic system. The molecular sub-systems of life have quite likely different capacities for realizing molecular organic codes (e.g. metabolism $<$ protein network $<$ gene regulatory network). And we hypothesize that the discovery of those reaction systems that provide easily many organic codes were important steps during the origin and evolution of life. Our approach should allow to make this hypothesis more precise and testable in the near future.

Acknowledgements. We thank the German Research Foundation (DFG) and the Jena School for Microbial Communication (JSMC) for funding.

References

1. Küppers, B.O.: Information and the Origin of Life. MIT Press, Cambridge (1990)
2. Shannon, C.E.: A mathematical theory of communication. The Bell Systems Technical Journal 27, 379–423, 623–656 (1948) (reprinted with corrections)
3. Arndt, C.: Information Measures: Information and Its Description in Science and Engineering. Signals and Communication Technology, vol. 1. Springer, Berlin (2004)
4. Yockey, H.P.: Information Theory, Evolution, and the Origin of Life. Cambridge University Press, New York (2005)
5. Anderson, D.R.: Model Based Inference in the Life Sciences. Springer, Berlin (2008)
6. Barbieri, M. (ed.): Biosemiotics: Information, Codes and Signs in Living Systems. Nova Science Publisher, New York (2007)
7. Barbieri, M. (ed.): The Codes of Life: The Rules of Macroevolution, 1st edn. Biosemiotics, vol. 1. Springer, Netherlands (2008)
8. Favareau, D. (ed.): Essential Readings in Biosemiotics. Biosemiotics, vol. 3. Springer, Heidelberg (2009)
9. Barbieri, M.: The Organic Codes: An introduction to Semantic Biology. Cambridge University Press, Cambridge (2003)
10. Barbieri, M.: Biosemiotics: a new understanding of life. Naturwissenschaften 95(7), 577–599 (2008)
11. Faria, M.: Signal transduction codes and cell fate. In: Barbieri, M. (ed.) The Codes of Life: The Rules of Macroevolution, pp. 265–283. Springer, Dodrecht (2008)
12. Gabius, H., Andre, S., Kaltner, H., Siebert, H.: The sugar code: functional lectinomics. Biochimica et Biophysica Acta-General Subjects 1572(2-3), 165–177 (2002)
13. Turner, B.M.: Cellular memory and the histone code. Cell 111(3), 285–291 (2002)
14. Turner, B.M.: Defining an epigenetic code. Nat. Cell. Biol. 9(1), 2–6 (2007)
15. Fontana, W., Buss, L.W.: 'The arrival of the fittest': Toward a theory of biological organization. Bull. Math. Biol. 56, 1–64 (1994)
16. Speroni di Fenizio, P., Dittrich, P., Ziegler, J., Banzhaf, W.: Towards a theory of organizations. In: German Workshop on Artificial Life (GWAL 2000), Bayreuth, April 5-7 (2000)
17. Szathmáry, E.: Coding coenzyme handles: a hypothesis for the origin of the genetic code. Proc. Natl. Acad. Sci. USA 90(21), 9916–9920 (1993)
18. Yarus, M., Caporaso, J.G., Knight, R.: Origins of the genetic code: the escaped triplet theory. Annu. Rev. Biochem. 74, 179–198 (2005)
19. Koonin, E.V., Novozhilov, A.S.: Origin and evolution of the genetic code: the universal enigma. IUBMB Life 61(2), 99–111 (2009)
20. Adami, C., Ofria, C., Collier, T.C.: Evolution of biological complexity. Proc. Nat. Acad. Sci. USA 97, 4463–4468 (2000)
21. Dittrich, P., Speroni di Fenizio, P.: Chemical organization theory. Bull. Math. Biol. 69(4), 1199–1231 (2007)

An Open-Ended Computational Evolution Strategy for Evolving Parsimonious Solutions to Human Genetics Problems

Casey S. Greene, Douglas P. Hill, and Jason H. Moore

Dartmouth Medical School, Lebanon, NH 03756, USA
{Casey.S.Greene,Douglas.P.Hill,Jason.H.Moore}@dartmouth.edu
http://www.epistasis.org

Abstract. In human genetics a primary goal is the discovery of genetic factors that predict individual susceptibility to common human diseases, but this has proven difficult to achieve because these diseases are likely to result from the joint failure of two or more interacting components. Currently geneticists measure genetic variations from across the genomes of individuals with and without the disease. The association of single variants with disease is then assessed. Our goal is to develop methods capable of identifying combinations of genetic variations predictive of discrete measures of health in human population data. "Artificial evolution" approaches loosely based on real biological processes have been developed and applied, but it has recently been suggested that "computational evolution" approaches will be more likely to solve problems of interest to biomedical researchers. Here we introduce a method to evolve parsimonious solutions in an open-ended computational evolution framework that more closely mimics the complexity of biological systems. In ecological systems a highly specialized organism can fail to thrive as the environment changes. By introducing numerous small changes into training data, i.e. the environment, during evolution we drive evolution towards general solutions. We show that this method leads to smaller solutions and does not reduce the power of an open-ended computational evolution system. This method of environmental perturbation fits within the computational evolution framework and is an effective method of evolving parsimonious solutions.

1 Introduction

Computational evolution (CE) is a promising open-ended evolution approach inspired by the intersection of artificial life approaches which mimic biology and artificial evolution approaches which draw inspiration from darwinian processes [1]. The goal of CE, as opposed to artificial evolution, is to provide an open-ended evolutionary framework which draws heavily from true biology. A primary benefit of this approach is that researchers may simultaneously discover good solutions to real problems and the evolutionary processes that led to those solutions [2]. The flexible CE approach has previously been applied to both artificial life

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 313–320, 2011.

problems with Avida-like organisms [3] and human genetics where the organisms are models predictive of disease risk [2,4]. Here we introduce a biologically inspired method capable of driving the system towards parsimonious solutions. Specifically we explore the impact of environmental noise on evolved solutions in a computational evolution system (CES) capable of building complexity.

We evaluated the effect of environmental noise using the CES developed by Moore et al. [4]. This CES, shown in figure 1, is capable of both evolution of Avida-like organisms and open-ended evolution for bioinformatics problem solving in the domain of human genetics. This framework is hierarchically organized and the version used here is described in detail in Moore et al. [2]. For this work

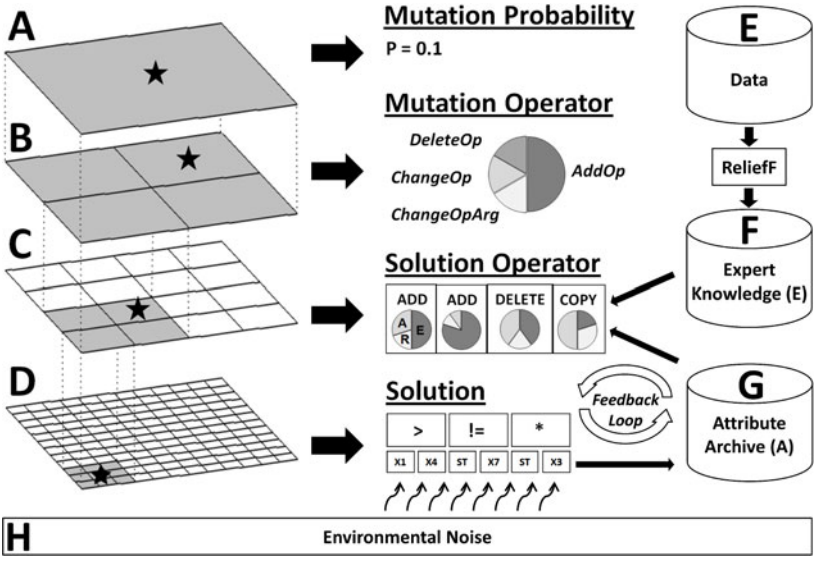


Fig. 1. An overview of the computational evolution system (CES) for discovering symbolic discriminant functions that differentiate disease subjects from healthy subjects using measurements of single nucleotide polymorphisms (SNPs). The hierarchical structure is shown on the left while specific examples are shown in the middle. At the lowest level (D) is a grid of solutions. Each solution consists of a list of functions and their arguments (e.g. X1 is an attribute) that are evaluated using a stack (denoted by ST in the solution). These solutions are applied to the dataset (E) at each generation and environmental noise (H) is added in various amounts. Here we examine the impact of environmental noise on solutions generated by the system. The next level (C) is a grid of solution operators that consist of some combination of the ADD, DELETE and COPY functions. These are capable of using ReliefF scores (F) or an attribute archive (G). ReliefF scores are derived by pre-processing the data (E). The attribute archive (G) is derived from the frequency with which each attribute occurs among solutions in the population. The top two levels of the hierarchy (A and B) exist to generate variability in the operators that modify the solutions. This system allows operators of arbitrary complexity to modify solutions. Note that we used 36x36 grids of 1296 solutions in the present study. A 12x12 grid is shown here as an example.

we modify the system by altering the data with environmental noise, shown as H. At each generation we begin with the initial dataset. Next we alter a specified portion of the SNPs for each individual. Because each SNP can have three potential genotypes, alteration consists of setting the current genotype for that SNP for that individual to a genotype value chosen randomly.

1.1 Robustness and Noise

Biological organisms evolve to be robust in response to a changing environment [5]. A species can be extremely successful in a narrow environmental range, but to survive environmental changes the species must continue to succeed under a variety of conditions. By analogy, a CES should evolve solutions that work well over a range of datasets, not just a single training set. This is particularly important for the genetic analysis of common human diseases where many believe genetic, environmental and stochastic factors play key roles in determining disease susceptibility [6]. The complex relationships between these factors lead to noisy datasets in which it is critical that over-fitting be prevented. Indeed, replication in an independent dataset has become a recognized criterion for validation of results in this field [7]. In biology, a species may become over-adapted if it is isolated for generations in an unchanging environment. Similarly, a CES may over-fit when trained on an unchanging dataset.

Here we propose a method to evolve robust solutions: environmental noise. Selection for robustness is difficult to accomplish directly in a constant environment, but the presence of environmental noise can drive the evolution of robust solutions [5,8]. Frequent environmental changes require solutions to be robust if they are to exhibit consistently high fitness. Here we expect this robustness to be exhibited through a decrease in solution size. Larger solutions often use more environmental information (SNPs and functions) to better classify individuals, but when environmental noise changes from generation to generation these larger solutions also have greater exposure to noise and can be subject to greater fluctuations in fitness. For this reason we hypothesize that there is some level of environmental noise that will allow solutions to evolve which are more compact (i.e. have fewer functions) and use fewer attributes (SNPs) than solutions evolved on a static dataset, while having similar or better predictive power on an independent dataset.

1.2 The Problem Domain

The practice of human genetics is rapidly changing due to the availability of new technologies that facilitate the measurement of more than 10^6 DNA sequence variations from across the genome. The field is no longer limited by ability to measure the genome but instead by a lack of tools to effectively analyze it. Current tools have had only limited success identifying genetic variations that play a role in the initiation, progression and severity of common human diseases such as breast cancer and schizophrenia [9,10].

Here we focus exclusively on one type of variation, the single nucleotide polymorphism (SNP). A SNP is a single point in the genome that differs between

people. The charge is to develop algorithms which can detect and characterize SNPs predictive of human health. Success is difficult due to non-linearity in the SNPs to disease mapping. This non-linearity is due, in part, to epistasis which is a term for non-additive gene-gene interactions. Epistasis is believed to be a ubiquitous component of the genetic architecture of common human diseases [11]. Therefore, the identification of genes with genotypes that confer an increased susceptibility to a common disease will require a research strategy that embraces this complexity [12]. The implication of epistasis for data mining strategies is that SNPs need to be considered jointly in learning algorithms and, because the mapping between the attributes and class is nonlinear, the concept difficulty is high [13]. Previously Moore et al. [2,4] have developed a computational evolution system (CES) capable of detecting and characterizing gene-gene interactions in human genetics. The goal here is to explore the role of environmental noise during fitness evaluation. Specifically we examine whether this noise allows this CES to effectively develop parsimonious models capable of explaining a complex non-linear relationship between genotype and disease.

2 Experimental Design and Data Analysis

Our goal was to evaluate this CES with and without environmental noise. This noise (Figure 1H) was added during solution evaluation. During each generation noise, H, was applied to the original data (E). For each SNP and individual combination there was a probability (h) that the observed genotype at that SNP would be changed to a different genotype. This dataset, now altered by environmental noise, was used to evaluate the fitness of the solutions. The solution best able to classify these altered data was compared to the best solution discovered previously on the original data (E). In this way we avoided choosing a best-of-run simply because it best classified the last generation of noise. It was this best-of-run which was returned as the solution. For the CES we used parameters from Moore et al. [2] (a solution grid size of 36x36, 500 generations, and a mutation frequency of 0.5). We examined the effect of various levels of environmental noise ($h = \{0.05, 0.1, 0.15, 0.2\}$) and compared results to those without noise ($h = 0$). A total of 100 runs with different random seeds were performed for each level of noise. By comparing solutions generated in many different runs which used varied levels of noise we could examine role of noise in evolution.

The central question addressed in this study is whether the presence of this noise causes evolved solutions to be more parsimonious. Here we evaluated parsimony through both the number of relevant functions and the number of unique attributes. The relevant functions in a solution were those that contributed to classification. Every function leaves its value on the stack and can take values from the stack; values that remained on the stack at the end of classification were not relevant. The attributes considered unique were those occuring at least once in a relevant portion of the solution. To determine whether different levels of noise led to statistically significant differences in solution parsimony we performed a Kruskal-Wallis (KW) analysis. KW is a non-parametric version of

the analysis of variance (ANOVA) which is appropriate when the assumptions of an ANOVA are not met [14]. This test allowed us to determine whether differences in the results obtained using various noise levels were likely to be due to chance. A significant KW p-value ($p \leq 0.05$) meant that one would only observe differences in the distributions of this magnitude one time out of twenty if there were no real effect. When the KW test was significant we performed a non-parametric post-hoc test. This post-hoc test allowed us to evaluate whether results for individual levels of noise ($h = \{0.05, 0.1, 0.15, 0.2\}$) were significantly different than results with no noise ($h = 0$).

We were also interested in whether noise significantly altered the power of the CES strategy. We therefore needed disease models, so we generated five models exhibiting complete epistasis consisting of two relevant SNPs with heritabilities of 0.4. This means that in these datasets approximately 40% of the phenotypic variability is explained by genotype. We used each of these models to simulate a training dataset and a validation dataset. To generate each dataset we simulated 1600 individuals (800 cases and 800 controls) using one of these models for the relevant SNPs. To force the CES to perform attribute selection we combined these relevant SNPs with 998 irrelevant SNPs for total dataset sizes of 1000 SNPs and 1600 individuals.

We examined how many times the CES discovered the correct SNPs and approximated a correct model. We evaluated the evolved solutions on the independent validation datasets. Across all models, solutions having the two relevant SNPs and an acceptable model using these SNPs could attain validation accuracies greater than 0.75 while those lacking either the SNPs or the model could not. We therefore considered any solution that attained an accuracy greater than 0.75 on these validation datasets correct. We counted the number of solutions that attained this or greater accuracy. This count, expressed as a percentage, was an estimate of the power of the method. We were also interested in the number of models that were unable to separate individuals in the validation datasets. These models had accuracies less than 45%, likely because they used rare cases to classify a small number of individuals in the training data and these situations did not occur in the validation data. We termed these invalid solutions and treated them similarly to power. To assess the reliability and robustness of these results quantitatively we use Fisher's exact test. Fisher's exact test is a significance test appropriate for categorical count data [14]. The p-value for this test could be interpreted as the likelihood of observing these differences in power without an association between amount of environmental noise and power. We considered results significant when $p \leq 0.05$. With this significance threshold we would only declare levels significant one time out of twenty when there was no real effect.

3 Results

Detailed results are shown in table 1. In order to address whether these solutions were more parsimonious we used two measures of solution complexity: the

Table 1. The results for all five models are summarized. Power, invalid solutions, the number of relevant functions, and the number of unique attributes are described in section 2. The numbers of relevant functions and unique attributes are summarized with means. Significance of differences from no noise ($h = 0$) are indicated by asterisks: * indicates $0.01 < p \leq 0.05$, ** indicates $0.001 < p \leq 0.01$ and *** indicates $p \leq 0.001$.

Model	Noise (h)	Power	Invalid Solutions	Relevant Functions	Unique Attributes
1	0.00	100	0	9.39	3.59
	0.05	100	0	7.79	2.61 (*)
	0.10	100	0	5.75 (***)	2.15 (***)
	0.15	98	0	6.28 (***)	2.26 (***)
	0.20	99	0	5.39 (***)	2.09 (***)
2	0.00	97	3	11.13	5.10
	0.05	100	0	5.74 (***)	2.56 (***)
	0.10	100	0	5.39 (***)	2.28 (***)
	0.15	100	0	4.81 (***)	2.18 (***)
	0.20	100	0	5.12 (***)	2.18 (***)
3	0.00	98	2	9.60	3.58
	0.05	100	0	7.12 (***)	2.46 (***)
	0.10	99	0	7.15 (**)	2.25 (***)
	0.15	100	0	5.93 (***)	2.16 (***)
	0.20	100	0	5.74 (***)	2.12 (***)
4	0.00	98	1	10.99	4.39
	0.05	100	0	8.12 (***)	2.77 (***)
	0.10	100	0	7.21 (***)	2.5 (***)
	0.15	100	0	5.7 (***)	2.15 (***)
	0.20	99	0	6.42 (***)	2.25 (***)
5	0.00	99	0	8.25	3.84
	0.05	100	0	5.58 (***)	2.49 (***)
	0.10	100	0	4.09 (***)	2.17 (***)
	0.15	99	0	4.42 (***)	2.11 (***)
	0.20	98	0	4.2 (***)	2.06 (***)

number of relevant functions and the number of unique attributes as described in section 2. Solutions evolved with greater amounts of environmental noise contained fewer relevant functions. The differences in the distributions observed were significant (KW $p \leq 0.05$). Additionally the post-hoc test showed that the number of relevant functions for each level of noise ($h = \{0.05, 0.1, 0.15, 0.2\}$) was significantly (post-hoc p-values≤ 0.05) different from the control number of relevant functions, i.e. $h = 0$ for all but model 1 with the lowest level of noise. Solutions with equivalent accuracy but fewer functions are often preferable because they are often easier to interpret and they take less computational time to evaluate.

We also observed that environmental noise reduced the number of unique attributes per solution. These differences were significant (KW p-value < 0.05) and the post-hoc test confirmed that the number of unique attributes for each level of noise ($h = \{0.05, 0.1, 0.15, 0.2\}$) was significantly different from the control ($h = 0$) number (p-values ≤ 0.05) across all models and levels of noise.

Solutions with successful functions that contained both relevant attributes achieved validation accuracies greater than 0.75 and were considered successful during the power analysis (see Section 2). The level of noise did not have a significant effect on the power of the CES approach for any of the individual models, which indicates that the increased parsimony did not appear to result in reduced power. Indeed across all models the power without noise (98.4%) significantly differed from the powers obtained with noise levels of 0.05 (100%, $p \leq 0.01$) and 0.10 (99.8%, $p \leq 0.05$) but did not significantly differ from the powers obtained with noise levels of 0.15 (99.4%, $p > 0.05$) and 0.20 (99.2%, $p > 0.05$). These results indicated that, at the tested levels of noise, power was not reduced and was actually significantly increased for some levels of noise.

Also of note was that, in rare cases, the solution evolved on training data was unable to classify individuals in validation data. This could occur when a genotype value used for classification in the training dataset did not occur in the validation data. When noise was used these rare situations could arise by chance which could allow for selection against non-robust solutions. Here we observed that invalid solutions (those unable to obtain an accuracy greater than 45% on the testing data) only occured when noise was absent. For each individual model differences were not significant but across all models the difference between the number of invalid solutions evolved in the absence of noise (6) was significantly different ($p \leq 0.05$) than the number of invalid solutions evolved in the presence of noise (0).

4 Discussion and Conclusions

We have shown that environmental noise is an effective biologically inspired method of evolving parsimonious solutions in an open-ended computational evolution system. Solutions involved in the presence of noise were consistently smaller and this decrease in solution size did not come at a cost of power. In fact for modest levels of noise the power was statistically significantly higher. Environmental noise is particularly attractive for CES approaches because of clear parallels to biology [5]. Our goal when using evolutionary computing in genetics is not to explain all of the training data but to discover underlying models that contribute to disease risk. By evolving solutions robust to environmental noise we avoid over-fitting the training data without reducing our ability to explain the underlying disease model. If these methods can reliably generate compact genetic models able to capture the complex interactions which predict individual susceptibility to common human disease, computational evolution and other artificial life inspired approaches will have a positive impact on our understanding of human health.

Acknowledgements

This work is funded by NIH grants LM009012, AI59694, HD047447, and ES007373.

References

1. Banzhaf, W., Beslon, G., Christensen, S., Foster, J.A., Kepes, F., Lefort, V., Miller, J., Radman, M., Ramsden, J.J.: From artificial evolution to computational evolution: a research agenda. Nature Reviews Genetics 7, 729–735 (2006)
2. Moore, J.H., Greene, C.S., Andrews, P.C., White, B.C.: Does complexity matter? artificial evolution, computational evolution and the genetic analysis of epistasis in common human diseases. In: Genetic Programming Theory and Practice VI, pp. 125–143. Springer, Heidelberg (2009)
3. Tyler, A.L., White, B.C., Greene, C.S., Cowper-Sal.Lari, R., Moore, J.H.: Development and evaluation of an open-ended computational evolution system for the creation of organisms with complex genetic architecture. In: Proceedings of the 2009 IEEE Congress on Evolutionary Computation, pp. 2907–2912 (2009)
4. Moore, J., Andrews, P., Barney, N., White, B.: Development and evaluation of an open-ended computational evolution system for the genetic analysis of susceptibility to common human diseases. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 129–140. Springer, Heidelberg (2008)
5. Wagner, A.: Robustness and Evolvability in Living Systems. Princeton University Press, Princeton (2005)
6. Aranda-Anzaldo, A., Dent, M.A.R.: Developmental noise, ageing and cancer. Mechanisms of Ageing and Development 124(6), 711–720 (2003)
7. Chanock, S.J., Manolio, T., Boehnke, M., Boerwinkle, E., Hunter, D.J., Thomas, G., Hirschhorn, J.N., Abecasis, G., Altshuler, D., Bailey-Wilson, J.E., Brooks, L.D., Cardon, L.R., Daly, M., Donnelly, P., Fraumeni, J.F., Freimer, N.B., Gerhard, D.S., Gunter, C., Guttmacher, A.E., Guyer, M.S., Harris, E.L., Hoh, J., Hoover, R., Kong, C.A., Merikangas, K.R., Morton, C.C., Palmer, L.J., Phimister, E.G., Rice, J.P., Roberts, J., Rotimi, C., Tucker, M.A., Vogan, K.J., Wacholder, S., Wijsman, E.M., Winn, D.M., Collins, F.S.: Replicating genotype-phenotype associations. Nature 447(7145), 655–660 (2007)
8. Felix, M.A., Wagner, A.: Robustness and evolution: concepts, insights and challenges from a developmental model system. Heredity 100(2), 132–140 (2008)
9. Shriner, D., Vaughan, L.K., Padilla, M.A., Tiwari, H.K.: Problems with Genome-Wide association studies. Science 316(5833), 1840–1841 (2007)
10. Williams, S.M., Canter, J.A., Crawford, D.C., Moore, J.H., Ritchie, M.D., Haines, J.L.: Problems with Genome-Wide association studies. Science 316(5833), 1841–1842 (2007)
11. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. Human Heredity 56, 73–82 (2003)
12. Tyler, A.L., Asselbergs, F.W., Williams, S.M., Moore, J.H.: Shadows of complexity: What biological networks reveal about epistasis and pleiotropy. BioEssays, 220–227 (2009)
13. Freitas, A.: Understanding the crucial role of attribute interactions. Artificial Intelligence Review 16, 177–199 (2001)
14. Sokal, R.R., Rohlf, F.J.: Biometry: the principles and practice of statistics in biological research, 3rd edn. W. H. Freeman and Co., New York (1995)

Gene Regulatory Network Properties Linked to Gene Expression Dynamics in Spatially Extended Systems

Costas Bouyioukos and Jan T. Kim

University of East Anglia, NR4 7TJ, Norwich, UK
k.bouyioukos@uea.ac.uk

Abstract. Gene expression levels within a cell are determined by the network of regulatory interactions among genes. In spatially extended systems of multiple cells, gene expression levels are also affected by activity in neighbouring cells. This interplay of a genetic regulatory network and interactions among neighbouring cells may qualitatively alter the dynamics of gene expression and is at the core of biological pattern formation.

In this study, we investigate the effects of the topology of a regulatory network on its pattern formation potential. We score networks by comparing the heterogeneity of gene expression levels generated on a lattice to that of the levels generated in a well stirred reactor as a null model, and assess the correlation of this score to characteristics of topology, such as density or centrality measures.

Density is strongly correlated to the potential to generate gene expression heterogeneity. For some networks that produce high heterogeneity on lattices, centrality and membership in cycles are indicative of the impact which deleting a gene has on the level of heterogeneity produced.

1 Introduction

Gene Regulatory Networks (GRNs) consist of genes and gene products. Each gene encodes a product, and a gene's expression level is the concentration of the product that the gene encodes. Gene expression rates, i.e. the speed at which a gene's product is synthesised, are controlled by regulatory interactions of gene products with the promoter of the gene [1, Ch.7]. Gene expression rates are subject to variation, activation of a gene increases its expression rate, and repression decreases its expression rate. The set of gene product concentrations within a cell at a given time constitutes the state of the cell. Following Kauffman's classical NK network approach [2], different cell types can be formalised as different stable states. The genes and the regulatory interactions of a GRN constitute its topology. The topology of a GRN plays a substantial role in controlling gene expression [3], and it also enables classification of GRNs, e.g. as a random graph or a scale free graph [4]. Cycles are prominent features of networks, and their role in determining dynamic properties of GRNs has been studied analytically by Thomas [5] and Soulé [6]. Quantitative models of GRNs need to include various

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 321–328, 2011.

dynamic parameters which describe properties of gene products such as decay and diffusion, as well as strength and other properties of regulatory interactions.

Many biological systems are spatially extended, e.g. tissues are spatially extended structures of cells. Gene expression levels are subject to variation within tissues, and neighbouring cells in a tissue exchange gene products through diffusion (and other processes). In such systems, the GRN (comprised of the topology and the dynamical parameters) is not sufficient to determine the dynamics of gene expression levels. Some processes, such as pattern formation, can occur only in spatially extended systems, and many aspects of dynamical systems may be impacted by spatial structure. The effect of spatial structure on hypercycle dynamics, studied by Boerlijst and Hogeweg [7], is a classical example of a qualitative effect of spatial structure on the dynamics of a chemical system.

Here, we explore correlations between the pattern formation potential of a GRN and its topological features (e.g. density, diameter, average clustering coefficient). Furthermore, we also investigate correlations between pattern formation capacity and individual gene characteristics (e.g. degree, centralities). As an indicator of pattern formation we consider heterogeneity of gene expression levels. We use a lattice as a spatially extended system. Lattices have been used in numerous studies to model multicellular systems, see e.g. [8,9]. We introduce an information based score, inspired by Maynard Smith [10], to quantify heterogeneity of gene expression levels, and employ an optimisation approach to find network topologies that have an elevated potential to generate heterogeneity in spatially extended systems. We determine the topological features of these GRNs and study their correlation to the information based score. To characterise individual gene effects, we simulate single gene knock-outs (i.e. loss of function mutations) by deleting each gene individually from the GRN. We then investigate correlations between the effects of these knock-outs on the score and individual gene measures such as centrality measures [11] and the number of cycles that a gene is a member of.

2 Methods and Analytical Framework

2.1 GRN Modelling

We use *transsys* [12], a computational framework to model GRNs. Transsys consists of a formal language to describe GRNs, a facility to simulate gene expression dynamics, and various other tools. A *transsys program* P represents a GRN and contains the declarations of factors and genes. A gene declaration consists of a promoter block and the product block. The product block contains the specification of the factor which the gene encodes. The promoter block consists of promoter elements, each of which specifies either constitutive expression or describes a regulatory effect of a factor on the expression rate of the gene. The parameters of such a regulatory element are a_{spec}, describing the binding specificity of the regulating factor to the element, and a_{max}, the maximum expression rate that the element can cause. The declaration of a factor f specifies

the decay rate r_f and the diffusibility d_f, a real valued parameter that represents the general ability of a factor to diffuse. The set of factors is denoted by $\mathcal{F}$. Gene knock-outs are straightforwardly simulated by deleting the gene in question from a transsys program. A *transsys instance* p of a transsys program P has a state which consists of the factor concentrations $C(f, p)$ for all factors $f \in \mathcal{F}$. Transsys operates in discrete time steps. Transsys instances provide an *update method* which uses the transsys program to compute the state at time step $t + 1$ based on the state of an instance at time t. Further information and a copy of transsys can be obtained from [13].

2.2 Spatial Organisation

Spatial structure is modelled by a 2D orthogonal lattice with periodic boundaries. Each lattice site is occupied by a transsys instance of the lattice's transsys program. A set of transsys instances $\mathcal{P}$ on a lattice is denoted by $\mathcal{P}_{\text{lattice}}$. In each time step, the amount of factor f that diffuses from an instance p to its 4 neighbours is given by $D_f = C(f, p) \cdot d_f/(4d_f + 1)$. The update of a lattice consists of the synchronous application of diffusion, followed by the invocation of the update method on each transsys instance on the lattice.

A well stirred reactor is implemented by simulating gene expression as in the lattice, but in addition after every update the positions of the transsys instances are randomised. This reactor has no spatial organisation and serves as the null model for our experiments. A set of transsys instances on a well stirred reactor is denoted by $\mathcal{P}_{\text{wellStirred}}$.

2.3 Quantifying Heterogeneity in Gene Expression

To quantify heterogeneity in factor concentration in a set of transsys instances a Shannon information based measure has been devised. A factor with homogeneous distribution of gene expression levels throughout a set of transsys instances is in maximum entropy state and contains no information, whereas factor distributions that exhibit heterogeneity have a positive information content. In a transsys instance p from a set of transsys instances $\mathcal{P}$, a factor f has relative concentration $R(f, p) = C(f, p)/C_{\text{total}}(f, p)$, where $C_{\text{total}}(f, p)$ is the sum of concentrations of factor f in $\mathcal{P}$. The Shannon entropy of this factor f on $\mathcal{P}$ is then given by:

$$H(f, \mathcal{P}) = -\sum_{p \in \mathcal{P}} R(f, p) \log_2 R(f, p) \tag{1}$$

A set of transsys instances where a factor's concentration is homogeneous has maximum Shannon entropy $H_{\max}(f, \mathcal{P}) = \log_2 |\mathcal{P}|$. The information content $I(f, \mathcal{P})$ of a factor f in the set $\mathcal{P}$ is $I(f, \mathcal{P}) = H_{\max}(f, \mathcal{P}) - H(f, \mathcal{P})$ For a set of transsys instances $\mathcal{P}$ of a transsys program with factor set $\mathcal{F}$, the information based measure for the whole set $I(\mathcal{P})$ is:

$$I(\mathcal{P}) = \sum_{f \in \mathcal{F}} I(f, \mathcal{P}) \tag{2}$$

2.4 Optimisation

An objective function was devised such that it returns negative values for networks that generate a higher level of heterogeneity on a lattice than in a well stirred reactor. For a lattice and a well stirred reactor populated with instances of a transsys program P, the same initial conditions (drawn randomly from the range of potential factor concentrations of the GRN) and both updated t times, the objective function is:

$$O(P,t) = I(\mathcal{P}_{\text{wellStirred},t}) - I(\mathcal{P}_{\text{lattice},t}) \tag{3}$$

We optimise the dynamic parameters of a transsys program by randomly perturbing them and accepting the perturbed parameters if the objective score is improved. The search process starts with randomly generated dynamic parameters. After a fixed number of rounds the optimised transsys program is returned.

2.5 Network Analysis

Network Properties. The following network based measures have been used to characterise GRNs:

- Clustering coefficient: a measure of the density of triangles in a network [14].
- Diameter: the length of the longest of all shortest paths in the graph [15, Ch.22].
- Number of cycles: the number of directed cycles in the graph.
- Average cycle length: the average length of the directed cycles of the graph.
- Density: the number of edges of the network over the maximum possible number of edges.

Each network measure was correlated with the objective score of each GRN after optimisation. Spearman's ρ index and the corresponding p-value was computed to quantitatively assess the correlations.

Individual Gene Properties. Node centrality measures have been employed to characterise individual genes based on the topology of a GRN. The degree, betweenness, closeness and eigenvector centrality as described in [11] were used. Moreover, to investigate the role of cycles, the number of cycles that each gene is member of has been computed. For each gene, the objective score of the single gene knock-out has been calculated and the loss in the objective score was correlated with each individual gene property. The *igraph* library [16] was used to compute most of the properties described in the network analysis section.

3 Results and Discussion

We use lattices of 5 cells height and 60 width. Both the lattice and the well stirred reactor were run for 400 time-steps to allow the systems to go through

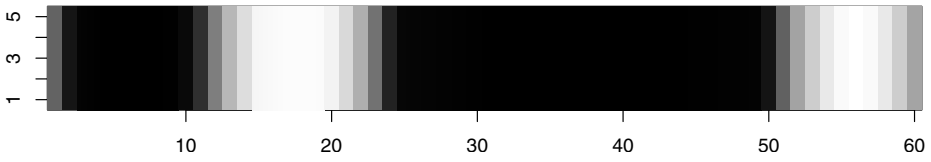


Fig. 1. Grey-scale image of factor concentrations of one factor on a lattice after optimisation. Concentration values range from ≈ 0 (black) to 0.27 (white).

initial transients and move towards attractors. Figure 1 shows a grey-scale image of the concentrations of a factor after optimisation on a lattice. This is a typical example of gene that shows a heterogeneous, striped pattern of expression levels on the lattice, whereas in the well stirred reactor expression levels are homogeneous. The information content of this factor f on the lattice is $I(f, \mathcal{P}_{\text{lattice},400}) \approx 1.3$ bits and the well stirred reactor $I(f, \mathcal{P}_{\text{wellStirred},400}) \approx 0$, the maximum information content for a factor on the lattices of our experiments is $H_{\max} = \log_2 300 \approx 8.2$.

3.1 Network Properties Results

We randomly generated networks of 15 nodes and 45 edges, using either the Erdös-Rényi (ER) method [17] or a process generating power-law (PL) degree distributions. We generated 15 networks of each type, thus obtaining 30 networks. Each network was optimised three times, starting from different dynamic parameter settings, thus producing 90 GRNs in total. Figure 2 shows correlation plots of network properties and the objective scores obtained after optimisation for the 90 GRNs. There is no detectable correlation of the average clustering coefficient and the number of cycles to the objective score ($p > 0.05$). A smaller diameter is weakly ($p = 0.017$) correlated to smaller objective scores and thus more heterogeneity on the lattice. Similarly, a smaller average cycle length is correlated to more heterogeneity on the lattice ($p = 0.028$).

3.2 Effects of Density

Random networks consisting of 15 nodes and with a number of edges ranging from 16 to 60 were used to investigate density effects. For each number of edges, 4 ER and 4 PL networks were randomly generated. Each network was optimised starting from 3 initial dynamic parameter settings, resulting in 552 GRNs. Figure 3 shows the results. There is a very strong correlation of density to the objective score ($p \ll 0.0001$). Higher density is related to lower objective scores, and at low density values, minimal scores are much closer to 0 than they are at higher densities.

3.3 Gene Characteristics Results

For each of the 90 GRNs used in Sec. 3.1 a full set of single gene knock-out experiments has been conducted. Results from a representative GRN which pro-

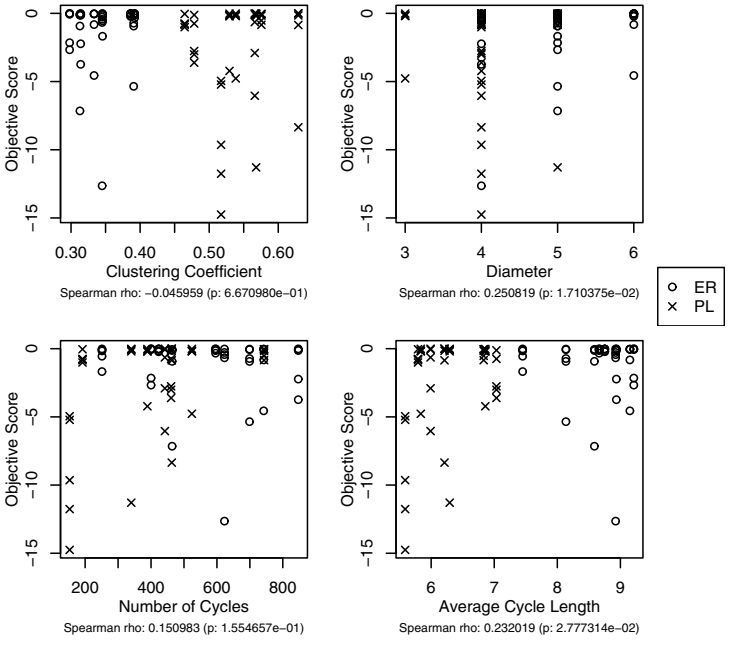


Fig. 2. Correlation plots of network topology properties against the objective scores of 90 GRNs

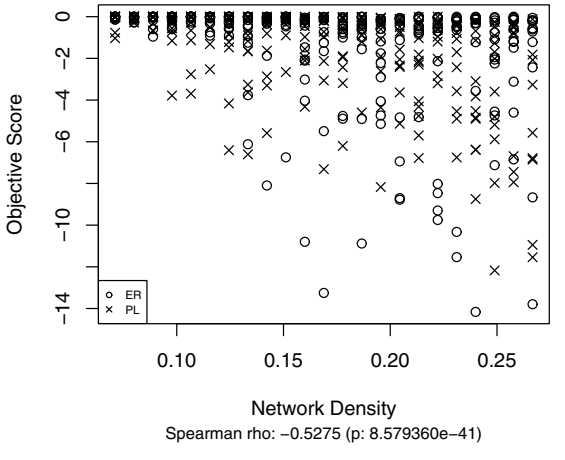


Fig. 3. Correlation plot of network density against objective scores of 552 GRNs

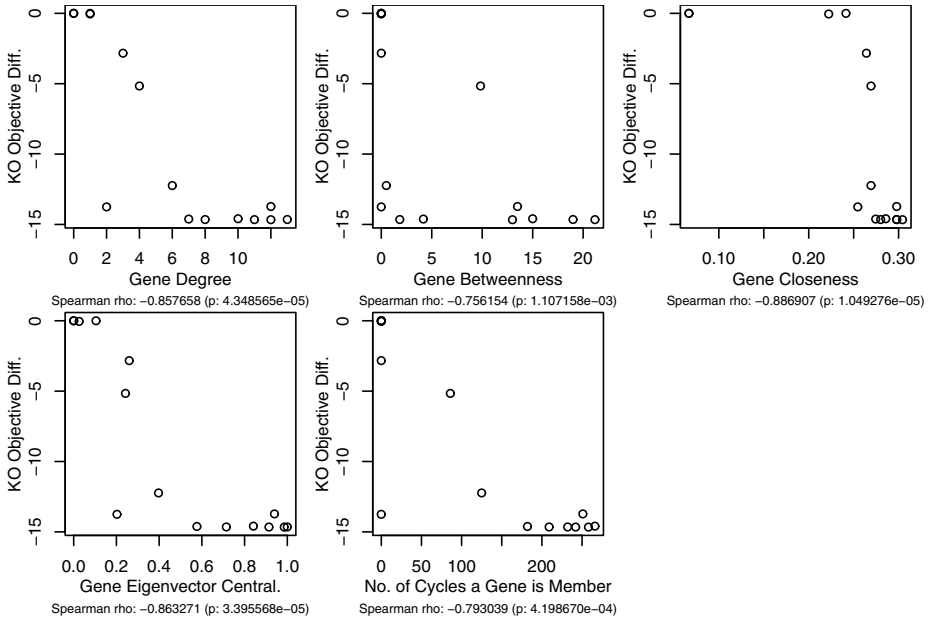


Fig. 4. Correlation plots from a GRN with low objective score. Four correlation plots depict centralities of individual genes against the loss in the objective score due to the gene knock-out (KO). The last is the number of cycles that a gene is member of against the same objective score loss.

duces a stripe pattern on the lattice are shown in Fig. 4. All centrality measures (degree, closeness, betweenness and eigenvector centrality) correlate strongly ($p < 0.0001$) with the objective score loss due to the knock-out. Loss in objective score is also correlated ($p \approx 0.0004$) to the number of cycles the knocked-out gene is member of. The more cycles a gene is member of the greater is the impact of the deletion of this gene in the objective score. This finding might be related to the roles that cycles play in controlling the factor concentration patterns in spatial systems and motivates further studies on the effect of cycles in the behaviour of GRNs.

3.4 Summary and Outlook

We have explored relationships between topological features of a GRN to its potential to take advantage of spatial structures to generate heterogeneity and patterns. With constant density, we have not found any global features of topology (diameter, number of cycles, average clustering coefficient, average cycle length) that are strongly correlated to the potential to generate heterogeneity of gene expression levels in a spatially extended system. Density itself is strongly correlated to such heterogeneous gene expression levels. For GRNs that produce spatial heterogeneity of gene expression, this property is highly related to

topological features of individual genes (i.e. centralities and the number of cycles that a gene is member of). As cycles play an important role in gene expression dynamics, we plan to focus our immediate future research on further characterising cycles and studying their impact on pattern formation.

References

1. Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., Walter, P.: Molecular Biology of the Cell, 4th edn. Garland Science, USA (2002), ISBN: 0815340729
2. Kauffman, S.A.: Developmental logic and its evolution. Bioessays 6(2), 82–87 (1987)
3. Barabási, A.L., Oltvai, Z.N.: Network biology: Understanding the cell's functional organization. Nature Reviews Genetics 5(2), 101–113 (2004)
4. Albert, R., Barabási, A.L.: The statistical mechanics of complex networks. Reviews of Modern Physics 74, 47–97 (2002)
5. Thomas, R.: Logical analysis of systems comprising feedback loops. Journal of Theoretical Biology 73(4), 631–656 (1978)
6. Soulé, C.: Graphic requirements for multistationarity. ComPlexUs 1(3), 123–133 (2003)
7. Boerlijst, M.C., Hogeweg, P.: Attractors and spatial patterns in hypercycles with negative interactions. Journal of Theoretical Biology 176(2), 199–210 (1995)
8. Bignone, F.A.: Cells-gene interactions simulation on a coupled map lattice. Journal of Theoretical Biology 161(2), 231–249 (1993)
9. Keränen, S.V.E.: Simulation study on effects of signalling network structure on the developmental increase in complexity. Journal of Theoretical Biology 231(1), 3–21 (2004)
10. Maynard Smith, J.: The, Crafoord Prize Lectures. The idea of information in biology. The Quarterly Review of Biology 74(4), 395–400 (1999)
11. Koschützki, D., Schreiber, F.: Comparison of centralities for biological networks. In: Giegerich, R., Stoye, J. (eds.) Proceedings of the German Conference on Bioinformatics (GCB 2004). LNI, vol. P-53, pp. 199–206 (2004)
12. Kim, J.T.: `transsys`: A generic formalism for modeling regulatory networks in morphogenesis. In: Kelemen, J., Sosík, P. (eds.) ECAL 2001. LNCS (LNAI), vol. 2159, pp. 242–251. Springer, Heidelberg (2001)
13. Kim, J.T.: The transsys home page (2009), `http://www.transsys.net/`
14. Watts, D.J., Strogatz, S.H.: Collective dynamics of "small-world" networks. Nature 393(6684), 440–442 (1998)
15. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press, Cambridge (2001)
16. Csárdi, G., Népusz, T.: The igraph software package for complex network research. Inter. Journal Complex Systems, 1695 (2006)
17. Erdös, P., Rényi, A.: On random graphs. Publicationes Mathematicae 6, 290–297 (1959)

Adding Vertical Meaning to Phylogenetic Trees by Artificial Evolution

Francesco Cerutti[1,2], Luigi Bertolotti[1,2],
Tony L. Goldberg[3], and Mario Giacobini[1,2]

[1] Department of Animal Production Epidemiology and Ecology,
Faculty of Veterinary Medicine, University of Torino, Italy
[2] Molecular Biotechnology Center, University of Torino, Italy
[3] Department of Pathobiological Sciences, School of Veterinary Medicine,
University of Wisconsin-Madison, USA
{francesco.cerutti,luigi.bertolotti,mario.giacobini}@unito.it,
tgoldberg@svm.vetmed.wisc.edu

Abstract. Phylogenetic trees are the most commonly used method for representing the relationships among living organisms. Additive trees are often used to show evolutionary features, based on models of molecular evolution. In this case, information in the tree is contained only in the root-to-node direction or, in other words, in its topology. Indeed, in a typical left-to-right phylogram, the vertical order of taxa is meaningless, and the degree of similarity between taxa is reflected by the branch path between them. In an effort to make unresolved trees more informative, we applied a (1+1) Evolutionary Algorithm to find the best graphical tree representation that includes vertical information. The order of taxa linked to polytomic nodes is defined using data from distance matrices created from different features of taxa, such as genetic, temporal or geographical data. In this way, the vertical ordering of taxa on a phylogenetic tree can be used to represent non-genetic features of interest.

1 Introduction

A central goal of evolutionary biology is to describe the 'Tree of Life', or to infer relationships among all living organisms. Phylogenetic trees are the most commonly used representations of these relationships, consisting of a combination of nodes connected by branches. Extant individuals are represented by terminal nodes (branch tips), linked together through a common internal node, which represents a common ancestor.

Additive trees, whose branches contain information about the degree of difference between nodes, are often used to show evolutionary features. Such trees are commonly based on genetic information and models of molecular evolution. In this case, information in the tree is contained only in the root-node direction, by the pattern of linkages between branches and nodes; or, in other words, in its topology. Indeed in a typical 'left-to-right' phylogram, like the one shown in Fig. 1, the vertical order of taxa is meaningless, and the degree of similarity between taxa is only reflected by the branch path between them [1].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 329–336, 2011.

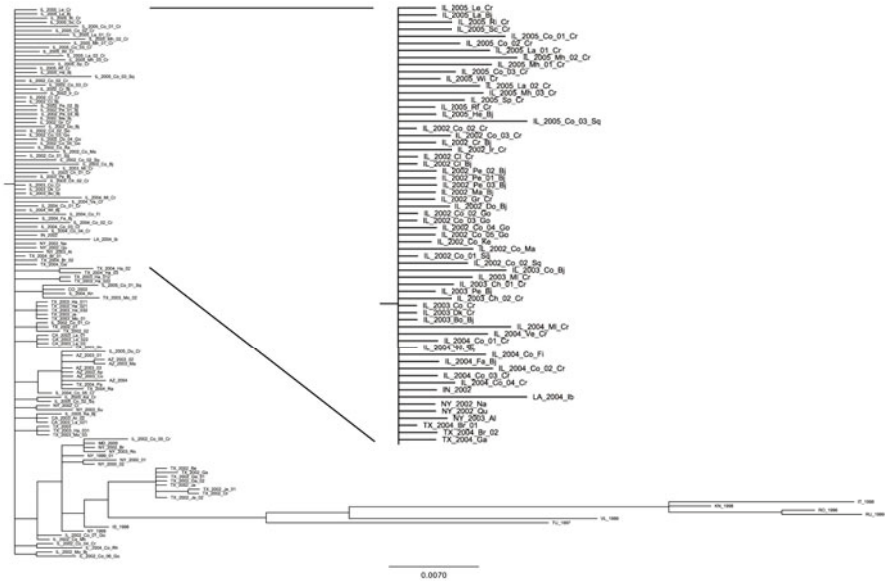


Fig. 1. West Nile virus phylogenetic topology obtained by the Bayesian approach MrBayes software and proposed in [2]

A second important feature of phylogenetic trees is the node's degree: in a fully resolved tree, all internal nodes have a degree equal to three, but, because of simultaneous divergences of sequences or, more likely, because of insufficient data, nodes can be a polytomies, joining more than three branches. In this last case, trees are hard to interpret. Often in such cases, trees are misinterpreted, with meaning mistakenly ascribed to the vertical proximity of taxa or clades. In cases where the vertical ordering of taxa on phylogenetic trees is flexible, the opportunity exists to ascribe biological meaning to this ordering [3].

In an effort to make unresolved trees more informative, we applied a heuristic search method to find the best graphical tree representation that would give biological meaning to the vertical ordering of taxa in a phylogenetic tree. Since, in a typical dendrogram or cladogram, each node can be freely rotated without changing tree's topology [1], it would be possible to group samples with similar features in the vertical direction, using any kind of relevant information. Taxa order in polytomic nodes could, for instance, be represented using data from distance matrices created from different samples features, as genetic, temporal or geographical data. Such an approach would allow one to draw more informative phylogenetic trees, especially when such trees contain unresolved nodes.

In the next section, we introduce and discuss the proposed heuristic search approach. Then, the method is experimentally validated using a West Nile virus phylogenetic tree in section 3. Finally, in section 4 we present our conclusions and discuss possible future work.

2 Searching for More Informative Trees

Finding the tree topology that better describe the phylogentic relations between a given set of taxa is a difficult problem, and the end result is often a 'best' tree that is only partially resolved.

For trees containing polytomies, taxa 'closeness' can be easily misinterpreted, leading to erroneous attribution of meaning to the vertical order of taxa. Given that each internal node can be freely rotated without changing tree's topology, it should be possible to find the best graphical tree representation that includes vertical information. This can be seen as an optimization problem. Its search space contains all the trees that can be obtained by node rotations from a topology previously obtained using other heuristic methods based on prior or posterior probability estimations. In this framework, the search problem would be to find the graphical representation that minimizes the distance between adjacent taxa, such distances being defined in matrices created from different sample features than were used to construct the tree, such as genetic, temporal or geographical data.

For phylogenetic trees containing large numbers of taxa, this optimization problem can not be solved by exhaustive searching, mainly because of the size of the search space. In fact, given a topology with N nodes, each with degree $\{d_1, d_2, \ldots, d_N\}$ (the root having one branch), the space S of all possible trees would have the size:

$$|S| = \prod_{i=1}^{N} (d_i - 1)!$$

In the case of a phylogenetic tree with 64 taxa, this dimension would range from $|S| = 2^{63}$ in the case of a completely resolved tree (thus containing 63 internal nodes all with degree 2), to $|S| = 64!$ for a completely unresolved tree.

Heuristic search method is therefore needed to solve this problem. The simplest approach would be to use a hill-climbing algorithm starting from the topology obtained by the tree-building method. Most such methods output a tree in which taxon order in polytomic nodes is often alphabetical or is taken from the order of input of sequences. A simple hill-climber would, however, be computationally unfeasible in most cases, since for each tentative solution s the number of neighbors $|neigh_s|$ to be generated and evaluated would be

$$|neigh_s| = \sum_{i=1}^{N} \binom{d_i - 1}{2} = \sum_{i=1}^{N} \frac{(d_i - 1)(d_i - 2)}{2},$$

where $\{d_1, d_2, \ldots, d_N\}$ are the degrees of the N internal nodes of the tree topology. Considering the same case as above of a tree with 64 clades, this value would range from $|neigh_s| = 64$ to $|neigh_s| = (64 \times 63)/2 = 2,016$.

As a first approach, we have decided to use a (1+1) Evolutionary Algorithm (EA). Starting from the original tree, in each generation a new tree is generated by applying a random swap between two taxa connected to the same node in the tree. The fitness of the new tree is evaluated as the sum of all the distances

between each node and the closest r tips according to the genetic distances matrix (r being the radius in the fitness evaluation, explained below). If the fitness of the new tree is better than that of the one in the previous generation, the new tree replaces the old one, and the search procedure continues. If not, the old tree is retained. This process is iterated for 200,000 generations, resulting in the creation and evaluation of 200,000 new tentative solutions.

When evaluating the fitness of a tentative solution, the most straightforward method would be to sum up, for all taxa in the tree, the distance between the taxon under consideration and the two taxa next to it, i.e. taxa at radius $r = 1$. However, it is not obvious that such a choice would be optimal since, when vertically reading a tree in the vertical dimension, one would also be inclined to consider as 'close' also taxa at distances greater than 1. To evaluate the influence of this parameter on the search dynamics, we have therefore included different radii in the fitness calculation.

3 Experimental Validation

West Nile virus (WNV; *Flaviviridae*; *Flavivirus*) is a single stranded, positive-sense RNA virus member of the Japanese encephalitis serocomplex that is transmitted primarily through the bite of infected mosquitoes. Because of the recency of its introduction into North America, it has been possible to study the phylogenesis and the evolution of the virus. Such studies of the virus in North America have tended to report highly unresolved trees [4,2,5]. In these cases, information on genetic, spatial or temporal clustering was not apparent, such that population substructure was investigated using different approaches.

In this preliminary study we used the tree presented by Bertolotti and colleagues in 2007, in its original form depicted in Fig. 1 as starting point for the (1+1)-EA.

The original tree has a total of 132 taxa and 28 internal nodes. The root node has 76 branches, among which 62 were directly connected to terminal taxa (as magnified in Fig. 1), pointing out that this part of tree is highly unresolved. Relating to the previous formulas, the resulting search space is $|S| = 1.749 \times 10^{137}$, an area too large to be explored with a exhaustive search. Furthermore, every tree has $2,975$ neighbors. Thus, a simple hill-climber approach would be too computationally intensive compared with a (1+1)-EA.

To calculate the fitness of each tentative solution, we used the matrix of genetic distances among samples, corrected with the best fit molecular substitution model (GTR+Γ+I) [6,1].

The trees were evaluated using different fitness radii: in particular, we used $r = 1, 4, 8, 32$. For each r, 50 independent runs of the (1+1)-EA have been performed, thus generating 50 trees with new tips vertical positions. As an initial comparison among trees generated with different r, we calculated the fitness of each tree as its relative fitness improvement (obtained as the ratio between the fitnesses of the considered tree and of the original starting tree). We then classified each run as Gold, Silver and Bronze, corresponding to 0.8, 0.85 and

Table 1. Relative fitness improvement: for each fitness radius the number of runs (hits) over the 50 executed is shown, as well as the mean generation together with its standard error at which the target was found

	$r = 1$		$r = 4$		$r = 8$		$r = 32$	
	hits	generation	hits	generation	hits	generation	hits	generation
Gold	0	N/A	38	$89,349_{\pm 7,906.3}$	50	$39,092_{\pm 4,770.1}$	0	N/A
Silver	26	$93,243_{\pm 7,178.2}$	38	$67,230_{\pm 9,409.3}$	50	$2,035_{\pm 160.0}$	0	N/A
Bronze	27	$61,810_{\pm 9,053.5}$	39	$41,309_{\pm 6,638.4}$	50	$714_{\pm 83.4}$	50	$3,239_{\pm 195.7}$

0.9 relative fitness improvement, respectively (the lower fitness is the better one). Statistics on this classifications can be found in Table 1: for each fitness radius the number of runs is shown over the 50 executed together with the mean generation $\pm$ standard error at which the target was found.

With $r = 1$, no tree was found that reached the Gold position, but more than half reached both the Silver and the Bronze classes. For $r = 4$ more than half of the runs reached the Gold position (thus also the Bronze and Silver ones). Apparently, better results were obtained with $r = 8$, since all the 50 runs reached the Gold class. When the radius $r = 32$ was used in the fitness evaluation, all the runs improved only until 0.8 relative fitness (i.e. Bronze position). Trees at $r = 8$ reached the Bronze class always in less than $1,000$ generations, while the other radii obtained this improvement only after $40,000$ generations in average.

In order to compare the trees obtained using different fitness radii, all evolved trees were evaluated with all the possible radii, from 1 up to 132 (the total number of the taxa). Fig. 2 depicts the median fitness improvements evaluated at all possible radii for the runs evolved using the 4 different fitness radii (with a focus on the range $[1, 20]$). It can be observed that the radii 8, 9, and 10 correspond to the region where all trees showed the largest relative fitness improvement. This, together with the observation in the above paragraph, suggests these radii as those exhibiting the best evolvability.

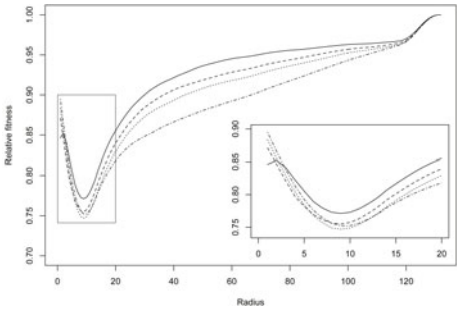


Fig. 2. Median fitness improvements evaluated at all possible radii for the runs evolved using the 4 different fitness radii (solid line: $r = 1$; dashed line: $r = 1$; dotted line: $r = 8$; dash-dotted line: $r = 32$). Box area highlights the radius range $[1, 20]$.

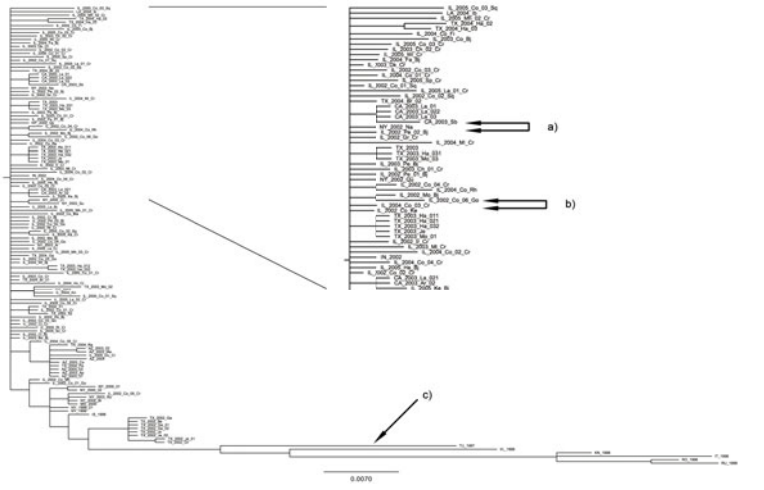


Fig. 3. A West Nile virus phylogenetic tree obtained using the proposed (1+1)-EA with fitness evaluation radius $r = 8$

In Fig. 2, it can be also observed that the evaluations of all trees converge to the same value as radius increases to approximately 120. In fact, when the radius is equal to the number of taxa in the tree, all trees in the search space show the same fitness. Thus, an intermediate value between those used to evolve them and the maximum (where all trees have the same fitness) should be chosen to compare the trees. We decided to use $r = 75$, a value that is well above that used in the search algorithm and that still allows discrimination between the different curves.

Statistical analysis of relative fitness improvements of the 4 different groups ($r = 1, 4, 8, 32$) evaluated at radius $r = 75$ shows a significant difference among the 4 groups (Kruskal-Wallis rank sum test $p < 0.001$). Moreover, these data highlight a significant ordering among the groups, having $r = 1$ worst than $r = 4$, $r = 4$ worst than $r = 8$, and $r = 8$ worst than $r = 32$ (all pairwise Wilcoxon rank sum test $p < 0.001$).

To justify the use of our fairly complex heuristic search method, we performed for comparison a random search algorithm with the same number of evaluations (n = 200,000). Briefly, our (1+1)-EA found trees with relative fitness improvement of 0.8344 (for $r = 1$), 0.7943 (for $r = 4$), 0.7478 (for $r = 8$), and 0.8511 (for $r = 32$), whereas the random search method found tree evaluated at 0.9065 (for $r = 1$), 0.8206 (for $r = 4$), 0.7699 (for $r = 8$), and 0.8878 (for $r = 32$). These results show that, for all radii, the evolutionary search method consistently outperforms random search.

The best final tree, shown in Fig. 3, has a completely different vertical taxon order than the original tree shown in Figure 2, reflecting genetic distances among the taxa. This presentation is able to highlight interesting features of the samples: for example, viruses that are from the same geographical origin tend to be

vertically close together (b), as are some viruses from different States, or collected in different period (a). Finally, the most genetically divergent group of viruses is moved to the bottom of the tree (c), highlighting this clade's uniqueness.

4 Conclusions and Future work

To make unresolved trees more informative, we applied a (1+1)-EA to find the best graphical tree representation that includes vertical information. Taxon order in polytomic clades was defined using data from distance matrices created from different samples features, as genetic, temporal or geographical data, in order to minimize the sum of the distances between adjacent taxa. To validate the proposed approach we applied it to the West Nile virus phylogenetic tree presented by Bertolotti and colleagues in 2007. This tree is highly unresolved, with a large number of samples belonged to highly polytomic internal nodes. To calculate the fitness value of each tentative solution we have employed the matrix of genetic distances among samples, corrected with the best fit molecular substitution model.

First, the influence of the radius for the fitness evaluation on the search dynamics was investigated. The trees under consideration were evaluated using different fitness radii ($r = 1, 4, 8, 32$). A random search with the same number of evaluations of the heuristic search was performed. Results show that our search method consistently outperforms the random one. The most informative results were obtained with $r = 8$, since all the runs reached top class in relative fitness improvement. This radius belonged to the region where all trees shown the largest relative fitness improvement.

In order to compare the trees obtained using different fitness radii, all evolved trees were evaluated at $r = 75$, a value that is well above that used in the search algorithm but that still allows one to discriminate between the different curves. Although, as expected, trees evolved using $r = 32$ showed improved fitness with respect to the other groups, the computational time needed to evolve these trees is long. Statistical analysis of the experimental data, together with the observations described above, suggest that a value for the radius in the fitness evaluation $r = 8$ might generally strike an acceptable balance between computational intensity and accuracy.

More generally, our results demonstrate that a heuristic search approach applied to tree graphical representation can improve the readableness of phylogenetic trees, helping in their interpretation. This being a preliminary study to validate a novel approach to add vertical meaning to phylogenetic trees, several issues need to be further investigated. Since the search dynamics of the proposed (1+1)-EA are influenced by the starting tree, future work should focus on different random initial tentative solutions. This study may also lead to future work incorporating real populational EAs, in particular $(1 + \mu)$-EAs and $(\lambda + \mu)$-EAs. Finally, our approach will need to be validated and applied to other case studies, and using other types of data from which distance matrices can be constructed.

Acknowledgements. The authors thank the Supercomputing Group of the CINECA Systems & Tecnologies Department for supporting in computations. M. Giacobini acknowledges funding (60% grant) by the Ministero dell'Università e della Ricerca Scientifica e Tecnologica. Luigi Bertolotti gratefully acknowledges financial support by Ricerca Sanitaria Finalizzata 2008 - Regione Piemonte. This work is supported by the National Science Foundation/National Institutes of Health Ecology of Infectious Diseases Program under award number EF-0840403.

References

1. Page, R.D.M., Holmes, E.C.: Molecular evolution: a phylogenetic approach. Blackwell Science, Oxford (1998)
2. Bertolotti, L., Kitron, U., Goldberg, T.L.: Diversity and evolution of west nile virus in illinois and the united states, 2002-2005. Virology 360(1), 143–149 (2007)
3. Maddison, W.: Reconstructing character evolution on polytomous cladograms reconstructing character evolution on polytomous cladograms. Cladistics 5(4), 365–377 (1989)
4. Davis, C.T., Ebel, G.D., Lanciotti, R.S., Brault, A.C., Guzman, H., Siirin, M., Lambert, A., Parsons, R.E., Beasley, D.W., Novak, R.J., Elizondo-Quiroga, D., Green, E.N., Young, D.S., Stark, L.M., Drebot, M.A., Artsob, H., Tesh, R.B., Kramer, L.D., Barrett, A.D.: Phylogenetic analysis of north american west nile virus isolates, 2001-2004: evidence for the emergence of a dominant genotype. Virology 342(2), 252–265 (2005)
5. Bertolotti, L., Kitron, U.D., Walker, E.D., Ruiz, M.O., Brawn, J.D., Loss, S.R., Hamer, G.L., Goldberg, T.L.: Fine-scale genetic variation and evolution of west nile virus in a transmission "hot spot" in suburban chicago, usa. Virology 374(2), 381–389 (2008)
6. Nei, M.: Molecular evolutionary genetics. Columbia University Press, New York (1987)

Transient Perturbations on Scale-Free Boolean Networks with Topology Driven Dynamics

Christian Darabos[1,2], Mario Giacobini[2,3], and Marco Tomassini[1]

[1] Information Systems Department, Faculty of Business and Economics
University of Lausanne, Switzerland
[2] Computational Biology Unit, Molecular Biotechnology Center
University of Torino, Italy
[3] Department of Animal Production Epidemiology and Ecology
University of Torino, Italy
{christian.darabos,marco.tomassini}@unil.ch, mario.giacobini@unito.it

Abstract. Taking into account the topology of genetic regulatory networks and abstracting recent findings about them, we investigate the behavior of a new, more biologically plausible, variation of the original Random Boolean Network paradigm. We study the dynamics of Boolean networks with scale-free structures, that evolve in time using a semi-synchronous topology-driven update scheme. Simulating statistical ensembles of networks, we discuss the attractors of the dynamics, and analyze in depth the fault-tolerance of the proposed model. Results are encouraging, as our model shows comparable and usually better performance and resilience to perturbations than the original one and is closer in spirit to real-life networks.

1 Introduction

Gene regulatory networks (GRNs) are extremely complex systems and we are just beginning to understand them in detail. However, it is possible, and useful, to abstract many biological details and focus on the system-level properties of the whole network dynamics. This Complex Systems Biology approach, although not strictly applicable to any given particular case, provides interesting general insight. Random Boolean Networks (RBNs) have been introduced by Kauffman more than thirty years ago [1] as a highly simplified model of GRNs. They have been studied in detail by analysis and by computer simulations of statistical ensembles of networks, and have been shown capable of surprising dynamical behaviors.

Today, we believe that Kauffman's original views are still valid, provided that the model is updated to take into account present knowledge about the topology of real GRNs without loosing its attractive simplicity. In a previous work [2] we have analyzed the dynamics of variations of the original RBN model that included some recent findings in the field of biological regulatory networks concerning the topological structure and the timing of events. In this work, we focus our efforts strictly on analysing the fault-tolerance of the model proposed

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 337–344, 2011.

above. Failures in systems can occur in various ways, and the probability of some kind of error increases dramatically with the complexity of the systems. Living organisms are robust to a great variety of genetic changes, and since RBNs are simple models of the dynamics of biological interactions, it is interesting and legitimate to ask questions about their fault-tolerance aspects. Kauffman [3] defines one type of perturbation to RBNs as "gene damage", that is the transient reversal of a single gene in the network. These temporary changes in the expression of a gene are extremely common in the normal development of an organism. The effect of a single hormone can transiently modify the activity of a gene, resulting in a growing cascade of alternations in the expression of genes influencing each other. This is believed to be at the origin of the cell differentiation process and possibly guides the development. Thus we analyze the behavior of the proposed variations of the original RBN model when subject to gene damage, a typical example of transient perturbation in networks dynamics.

In the next section we briefly review the main assumption of Kauffman's RBNs and their possible limitations. Changes to both the topology and the synchrony will be proposed in section 2. We introduce the concept of perturbation in section 3 and we investigate numerically the stability of the model. Finally, section 4 presents our conclusions and discuss possible future work.

2 Generalized Boolean Networks

Generalized Boolean Networks (GBNs) we have proposed in [2], are an extension of Kauffman's original RBNs model that aims at closing some known gaps between the original model and recent findings in biology. In RBNs and GBNs, the N vertices represent the Boolean *on/off* state of N genes' expression. But instead of each gene receiving K randomly chosen inputs from other genes, as it is the case in RBNs, GRNs adopts a scale-free topology [4,5] where the degree distribution follows a power-law $p(k) \sim k^{-\gamma}$ as proposed by Aldana [6], these specific GBNs are scale-free Boolean networks (SFBNs). Each gene's next state is decided by a randomly generated Boolean function of its inputs. The network dynamics is discrete and instantaneous in both RBNs and GBNs, with the difference that in GBNs the update sequence is neither fully synchronous (SU), nor asynchronous [7,8,9]. Instead, it is decided by the activation sequence of the genes. This update scheme is called Activated Cascade Update (ACU) and contrasts with classical RBNs fully synchronous update. In both cases, we start from an initial configuration where a random value is assigned to each gene. Simultaneously all the nodes in the case of SU and and a subset of nodes pointed by active genes for GBNs simultaneously examine their inputs, updating genes evaluate their Boolean functions, and find themselves in their new states at the next time step. As the systems are fully deterministic, they will travel through configurations, eventually looping back and cycling through a subset of the 2^N configurations called an attractor. It has been found that, as some parameters are varied such as the RBN's connectivity K, GBN's γ exponent, or the probability p of expressing a gene in the Boolean function, the systems can

go through a phase transition. Indeed, for every value of p, there is a critical value of connectivity K (for RBNs) or γ (for SFBNs) such that for values of K below this critical value the system is in the ordered regime, while for values above this limit the system is said to be in the chaotic regime. In the ordered regime, the distribution of perturbations sizes in the networks is a power-law with finite cutoff that scales as $\sqrt{N}$. Thus perturbations remain localized and do not percolate through the system. This tendency in inverted in the chaotic regime. Kauffman's hypothesis that living organism cells operate in a region bordering order and chaos, a "critical" regime where the dynamic conditions of the systems offers a tradeoff between stability and evolvability. In a previous work [2], we have studied in great detail the state space, attractor lengths, numbers, and distributions, comparing and contrasting results in all three regimes for both SU and ACU in both RBNs and SFBNs.

3 Transient Perturbations on Boolean Networks

In order to study damage spreading in dynamical Boolean networks, we have allowed samples of RBNs and GBNs under both SU and ACU to reach an attractor. In each case, the samples are made of 50 different topologies, each with 20 sets of distinct update functions and presented with 500 different initial configuration of the genes at time $t = 0$. Then we have submitted all systems that have reached biologically plausible attractors (up to 100 states) to "gene damage". The effect of gene damage can be measured by the size of the "avalanche" resulting from that single gene changing its behavior from active to inactive or vice-versa. The size of an avalanche is defined as the number of genes that have changed their own behavior at least once after the perturbation happened. The perturbed system is then compared to an unperturbed version that is running in parallel. That is, when the system is cycling through the configurations of the attractor, the whole system is duplicated. On the one hand, the original will continue unperturbed. On the other hand, a node of the copy is chosen at random and will give the opposite output value for a single time-step. This could potentially knock the system out of the course of its attractor. Then, we let both systems evolve over time and record at each time step how many more nodes have a different value in the copy compared to the original. This sum usually reaches a maximum that represents the number of nodes that have ever had a different behavior different with respect to that of its counterpart in the original system. This number is the size of the avalanche. There are only three possible scenarios for the copy: it will return to the same attractor as the original, reach a different attractor, or diverge and reach no attractor within the maximum number of configurations allowed (1000). Each system in an attractor is copied 10 times, and each copy will have a different avalanche starting point. We record separately these informations in order to compare the re-convergence capabilities of the systems in each regime, with different topologies and update schemes.

Fig. 1 shows the frequency at which systems that have already converged to an attractor do re-converge to one. We show separately wether systems re-converge

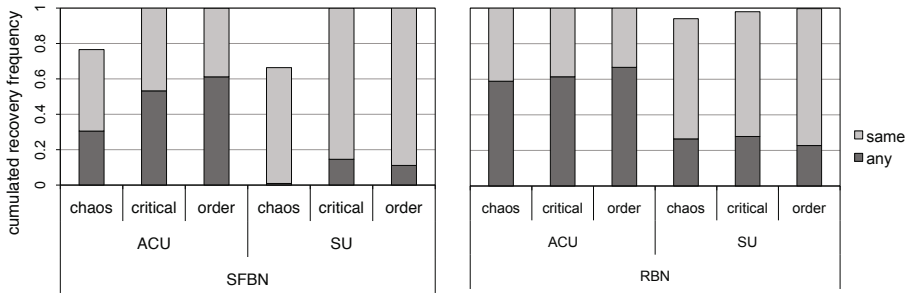


Fig. 1. Frequency at which systems re-converge to either the same attractor (light grey) or another one (dark grey). Left-hand side figure shows results for SFBNs and right-hand side figure shows results for RBNs. All systems have $N = 200$ nodes. We purposefully omit point attractors.

to the same one as before the perturbation or to a different one. In particular, Fig. 1 depicts results for attractors before perturbation (original attractors) of sizes between 2 and 100. We show networks of size $N = 200$ as results for smaller systems are comparable and are therefore not shown here.

Re-convergence mostly depends on the regime the system evolves in, rather than its degree distribution, update scheme, or size. In Fig. 1 we note that only networks in the chaotic regime do not re-converge to an attractor in every case. ACU performs a little better at helping systems to find a stable state. However, this tendency is inverted when taking into account only cases where the same attractors are found. In this case, under ACU, the same attractor as the original one is found about half of the time. Under SU, the same one is found about 75% of the time. This can be explained by the fact that the number of attractors lying in the states space of systems under ACU is much larger.

The size of the avalanche is directly related to the regime in which the RBN evolves; in the ordered regime, the cascades tend to be significantly smaller than in the chaotic regime. In biological cells, where the regime is believed to lie on the edge of chaos, the cascades tend to be small too. Moreover, the distribution of the avalanche sizes in the ordered regime follows a power-law curve [3] (see Fig. 3), with many small avalanches and few large ones. In the chaotic regime, in addition to the power law distribution, 30-50 precent of the avalanches are huge. The distribution of avalanches size of RBNs in the ordered regime roughly fits the expectations of biologists, where most of the genes, if perturbed, are only capable of initiating a very small avalanche, if any. Fewer genes could cause bigger cascades, and only a handful can unleash massive ones. Perturbing an arbitrary gene is reasonable in RBNs where all genes have the same average number of interactions. In scale-free nets however, this is no longer true due to the presence of a high degree-inhomogeneity. Even for values of γ around 2.5 there will be nodes that have many more output connections than the average value. A transient perturbation of a gene that has few interactions will have moderate or no effect, while perturbing a highly connected node will have larger consequences.

As expected when dealing with random failure, the information traveling through a structure with regular output distribution is more vulnerable to faults compared to structure with *hubs* and *leaves*. This fact is well known in various examples such as computer networks which are very resistant to failure as long as they are random and not targeted attacks on highly interconnected nodes. Especially under SU, SFBNs tend to re-converge to the same attractor more than RBNs, although overall, both topologies perform well. The chaotic case will be explained below in details. Under ACU, critical and ordered SFBNs systems are again performing as well as or better than their counter parts in RBNs, recovering as often to another attractor but more often to the same as the original one. The counter-performance of chaotic systems, especially SFBNs, can be explained by the "spike of huge avalanches" described by Kauffman [3] and visible in Fig. 3. Indeed, SFBN systems and, in a lesser manner, RBN under SU have a surge of very long avalanches when in the chaotic regime. This characteristic explains why these systems are not as performant at re-converging to an attractor, let alone the same one.

Fig. 2 shows the distribution of the avalanche sizes. Again, we distinguish networks that have re-converged at all in Fig. 2-left column and those that have re-converged to the original attractor Fig. 2-right column. As mentioned above, the size of the avalanche varies mainly due to the regime. Smaller systems with $N = 100$ react as expected, with the size of their avalanches increasing as the systems grows chaotic. However, this does not seem to always be the case,

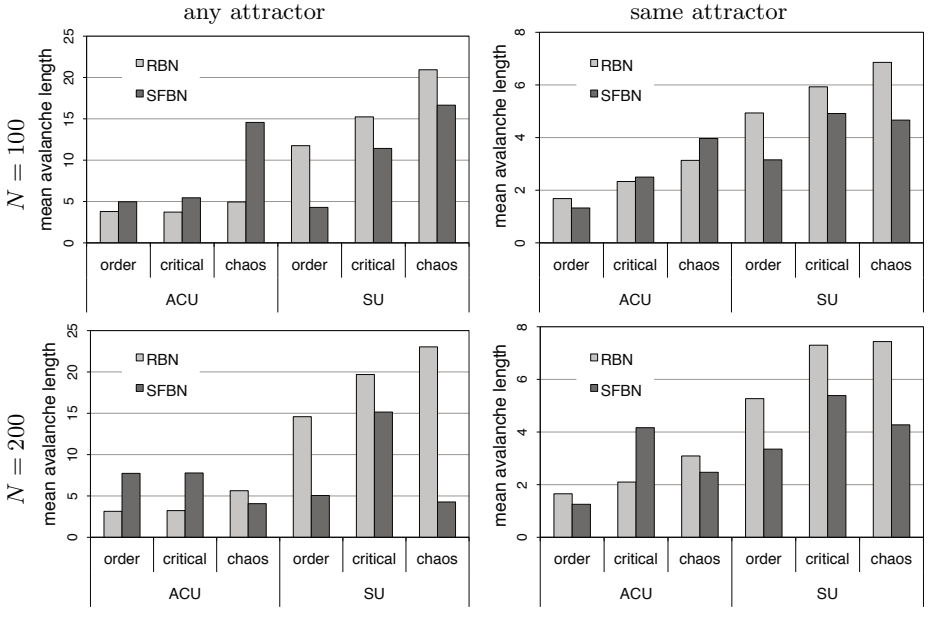


Fig. 2. Average avalanches length of cases where systems re-converge to any attractor (left) and the same attractor (right). In the upper row, results for network with $N = 100$, while those for $N = 200$ are depicted in the lower row.

and this relationship between avalanche size and regime is changed in bigger networks. Under ACU networks with $N = 200$, it is the systems that evolve in the critical regime that clearly show the longest avalanches. This is true for ACU only, SU systems still corroborate Kauffman's conjecture. Although in the case where systems return to the original attractor, avalanche sizes are much smaller, the tendencies observed in the more general case stand. This is the first time we observe an obvious impact of the networks size on the systems dynamics. Further investigations are necessary to define why larger systems in critical regime under ACU are more impacted by perturbations.

In Fig. 3, we show the distribution of the avalanches' sizes for different systems. Although values are discrete, we used continuous lines as a guide for the eye. We see that the tendencies are the same and are as anticipated from Kauffman's work [3]. SFBNs under both SU and ACU exhibit a steady long tailed decrease in the number of avalanches as their length grows for ordered and critical regime, and there is a surge of long avalanches in the case of chaotic systems. This is respected for synchronous RBNs in under SU. Interestingly, this does not to apply to RBNs under ACU, where no increment is to be noted.

Lastly, Fig. 4 illustrates the average output degree of the node representing the damaged gene. For clarity reasons, we show results only for bigger systems as they are similar when networks are scaled down.

Although predictable, we clearly see the effect of the hubs in SFBNs, where failing nodes in systems that do not re-converge have a much higher output

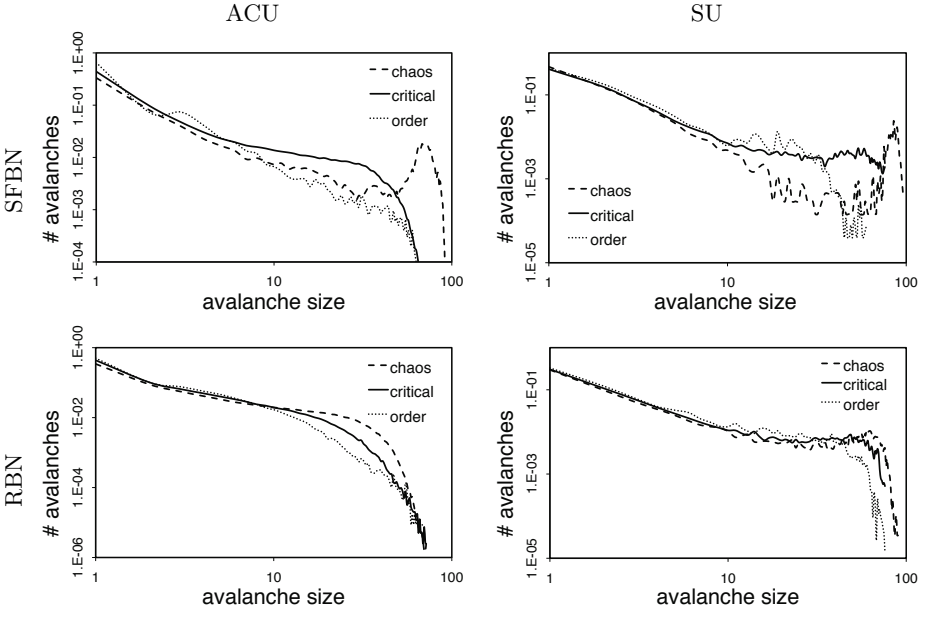


Fig. 3. Distribution of the avalanches lengths for all three regimes and $N = 100$. Upper row are SFBNs, lower row are RBNs. Left-hand side column are systems under ACU and right-hand side column, systems under SU.

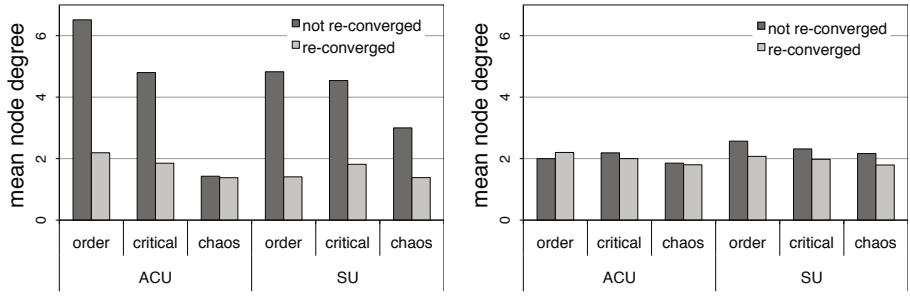


Fig. 4. Average degree of the nodes that have failed for both re-converged and not re-converged avalanches. On the right hand side RBNs are shown and SFBNs are on the left. Size $N = 200$.

degree in average than those of systems that did recover. Another interesting observation, is that there seems to be a direct relationship between the degree of the wrongful node and the regime, the more ordered the system, the higher the degree to allow the system to recover. This difference is toned down in ACU systems. Naturally this does not hold for classical RBNs, where all nodes have the same output degrees.

As a general conclusion on failures of Boolean Networks, we can highlight the prominent effect of the topology on distribution of the lengths of the avalanches and its ability to re-converge to an attractor over the networks update and regime.

4 Conclusions and Future Work

We are a long way from being able to build a model of GRNs that could help us understand the detail of the complex interactions that are taking place between the different components and with the external environment. Nevertheless, we are in the process of discovering what structural and dynamical properties make abstract models of GRNs highly stable and resistant to perturbation, and yet adaptable to mutation. This work, together with the previous one [2], identifies one structural property, namely the scale-free output distribution, and a dynamical one, the semi-synchronous updating, that try to account for recent findings in life sciences and use computer simulations to reflect the impact of these changes on RBN models. Results are encouraging, as our model shows comparable and usually better performances than the original one with more attractors and smaller avalanches. This leads us to believe that we are pointing in the right direction, trying to improve known models with present day knowledge. In the future, we intend to expand the range of analysis conducted on perturbed systems by introducing other common types of failures, in the hope of shedding some light on GRNs. Also, we would like to explore different degree distribution types and combination found in partially known real-life GRN. Today, this is possible thanks to high-throughput molecular genetics methods, which are making real-life data available like never before.

Acknowledgements

M. Tomassini and Ch. Darabos gratefully acknowledge financial support by the Swiss National Science Foundation under contract 200021-107419/1. M. Giacobini acknowledges funding (60% grant) by the Ministero dell'Universitá e della Ricerca Scientifica e Tecnologica.

References

1. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22, 437–467 (1969)
2. Darabos, C., Giacobini, M., Tomassini, M.: Semi-synchronous activation in scale-free boolean networks. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 976–985. Springer, Heidelberg (2007)
3. Kauffman, S.A.: Investigations. Oxford University Press, New York (2000)
4. Vázquez, A., Dobrin, R., Sergi, D., Eckmann, J.-P., Oltvai, Z.N., Barabàsi, A.-L.: The topological relationships between the large-scale attributes and local interactions patterns of complex networks. Proc. Natl. Acad. Sci. USA 101(52), 17940–17945 (2004)
5. Christensen, C., Gupta, A., Maranas, C.D., Albert, R.: Inference and graph-theoretical analysis of Bacillus Subtilis gene regulatory networks. Physica A 373, 796–810 (2007)
6. Aldana, M.: Boolean dynamics of networks with scale-free topology. Physica D 185, 45–66 (2003)
7. Harvey, I., Bossomaier, T.: Time out of joint: attractors in asynchronous random boolean networks. In: Husbands, P., Harvey, I. (eds.) Proceedings of the Fourth European Conference on Artificial Life, pp. 67–75. The MIT Press, Cambridge (1997)
8. Mesot, B., Teuscher, C.: Critical values in asynchronous random boolean networks. In: Banzhaf, W. (ed.) ECAL 2003. LNCS (LNAI), vol. 2801, pp. 367–376. Springer, Heidelberg (2003)
9. Gershenson, C.: Updating schemes in random Boolean networks: Do they really matter? In: Pollack, J. (ed.) Artificial Life IX Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, pp. 238–243. MIT Press, Cambridge (2004)

Agent-Based Model of Dengue Disease Transmission by *Aedes aegypti* Populations

Carlos Isidoro, Nuno Fachada, Fábio Barata, and Agostinho Rosa

Evolutionary System and Biomedical Engineering Lab
Systems and Robotics Institute
Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
{cisidoro,nfachada,fbarata,acrosa}@laseeb.org

Abstract. This paper presents an agent based model of the *Aedes aegypti* mosquito showing not only population dynamics but also the Dengue disease propagation in both the vector and host populations (mosquitoes and humans, respectively); this study will focus on the latter aspect. The agents model the main aspects of the mosquito's ecology and behavior, while the environmental components are implemented as a layer of dynamic elements obeying to physical laws. Model verification was performed through examination of simulation parameters variation and qualitative assessment with existing models and simulations. The agent based modeling and simulation platform used was the LAIS simulator.

Keywords: Artificial Life, Agent Based Modelling, Aedes aegypti, Dengue, RIDL, SIT.

1 Introduction

The dengue is a dangerous disease which still lacks a cure, and it is spread through a specific type of vector, the *Aedes aegypti* mosquito. Currently, the most affected areas are the ones with tropical climates since factors like high temperature and frequent precipitation are favorable to *Aedes aegypti* growth. However, if current predictions about climate change happen, many new areas might start facing the dengue threat [1].

Since an effective treatment is yet to be found, it is particularly important to focus on prevention, keeping the mosquito population under transmission threshold, or better still, eradicate the disease. Various strategies have been developed and used for this purpose, ranging from releasing large amounts of sterile mosquitoes into the environment to clearing areas with still water that might be used as mosquito breeding sites.

This paper is a continuation of [2], which studied the mosquito population dynamics and the effects of a particular population control strategy, RIDL. The study presented here will focus on the disease itself, more specifically on the

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 345–352, 2011.

propagation of the disease in both the mosquito (vector) and human (host) populations. These studies were performed using an agent based model, developed for the LAIS simulator.

ABM is well suited for describing complex systems in general and disease transmission in particular, providing a way to represent the true diversity of intervening components, such as environmental factors, disease vectors and disease hosts. Other advantages include the possibility to determine spatial behavior distribution, rapid insertion of new components and natural consideration of non-linear interactions between agents. This approach is not without problems of its own: it requires considerable computational power to simulate individual agents; parameter tuning is not trivial; and it lacks the formalism provided by differential equations (although ABM formalism is already a reality [3]). Nonetheless, for explicitly spatial models, such as the one presented here, the advantages of ABM clearly outweigh its limitations.

The state of the art in the modeling and simulation of the *Aedes aegypti* mosquito, dengue transmission and other relevant related subjects is presented in section 2; a brief description of the LAIS simulator is given in section 3, while the model itself is described in section 4. Sections 5 and 6 present the performed simulations, and the associated discussion, respectively.

2 State of the Art

There have been numerous models of mosquitoes and mosquito-borne disease, beginning with the classic Ross-Macdonald malaria models [4,5,6] and extending to present day models of vectors populations or aspects of vector biology, not directly considering disease [7,8,9,10].

One example of modeling the dengue vector mosquito population dynamics is by Focks and colleagues [11,12], examining the biology of *Aedes aegypti*. This is an exceptionally detailed model, with numerous types of containers for larval development. Hydrology (water levels and drying), temperature-dependent larval development, food availability and survival are explicitly tracked in each container type. Detailed weather data are used to drive the hydrological and biological functions. This level of detail has both costs and benefits; it enables consideration of detailed aspects of the mosquito biology, but also makes true sensitivity analysis of the model difficult or impossible. Thus, to develop a model with this level of detail, it is necessary to have extensive data available for parameter estimates and validation.

The use of ABM methodologies to model *Aedes aegypti* populations has been scarse at best. Some interesting ideas are presented in a work by Deng *et. al* [13], namely the use of an utility function to determine mosquito movement, taking into account factors such as population, wind direction, land use type and landscape roughness. However the practical implementation of the model is very limited, with coarse spatial discretization (30x30) and not singular agent-based.

Models can be useful to evaluate different strategy of mosquito control. Recently, techniques like releasing genetic modified mosquitoes have been considered as an enhanced SIT to control the mosquito population, as the genetic manipulation in insects result in sterility or lethal genes[14,15]. Although there wasn't any genetic modified mosquito open field release conducted yet, a couple of mathematical modeling works have been done to assess the control efficacy [16,17,18]. But none of those could provide a tool to simulate the interaction between mosquito individuals such as mating behavior, spatial distribution, and immigration etc. All these are important for the evaluation and guidance of genetic control approach.

3 The LAIS Simulator

The LAIS simulator is a multithreaded agent-based simulation platform, offering a modeling paradigm and a set of tools for the simulation of complex systems [19]. The platform is implemented in Java and makes use of several open source libraries which provide tools for spatial organization and visualization, event scheduling, simulation output (e.g., charts, CSV files, movies) an d simple class development and instantiation using XML. Simulations are performed in discrete time and two-dimensional discrete space. As such, space is divided into blocks, which are independently processed by different threads, making LAIS scalable on modern multiprocessor systems.

There are two main actors in the LAIS framework: *agents* and *elements*. Agents are typical ABM discret e and independent decision-making entities. When prompted to act, each agent analyzes its current situation (e.g. what resources are available, what other agents are in the vicinity), and acts accordingly, based on a set of rules. These rules incorporate knowledge or theories about the respective low-level components. On the other hand, elements are real-valued objects which obey predetermined rul es, such as physical laws (e.g., diffusion).

4 Model Description

The *Aedes aegypti* LAIS model implements a square topology where each spatial block has 8 neighbors (N,NE,E,SE,S,SW,W,NW). Five different agents are considered: Wild Male Mosquitoes (WM), Female Mosquitoes (WF), Sterile Male Mosquitoes (SM), Humans (H) and Oviposition spots (OS). Various different elements are also used, with the most important falling into one of the following categories: mosquito attractors and mosquito density measure.

The interactions between the various agents are represented in a simplistic way in fig. 1; a brief explanation follows:

WM-WF. WM are attracted to the pheromone released by WF. If a WM and a WF are on the same cell, there is a chance for the WF to become fertilized.

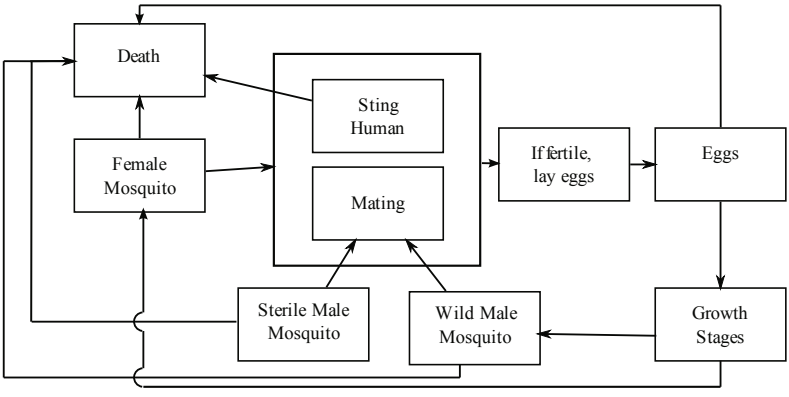


Fig. 1. Model overview

SM-WF. SM are also attracted to the pheromone released by WF. If a SM and a WF are on the same cell, there is a chance for the WF to register having mated, although it has not been fertilized (the implications will be given on the WF-OS interaction).

WF-Hu. WF follow the Humans body heat, and if they are in the same cell as a human they have a chance to either die or successfully acquire human blood.

WF-OS. After having mated and having acquired human blood, a WF will move towards an OS by following the humidity released by them. After reaching an OS, the WF will lay a certain number of eggs if it mated with a WM, or lay none if its mate was a SM. Afterwards it will again start looking for mates and humans.

Mosquitoes only interact with other agents after they mature into adults. After hatching from their eggs, they go through a number of development stages before becoming adults.

It is important to note that the elements used as mosquito attractors, are intended to model mosquito behavior and might not correspond to the exact process the mosquitoes use to follow their targets.

While not present in figure 1 and on the explanations given above, the spread of the Dengue disease can happen during the WF-Hu or the WF-OS interactions. In the first case (WF-Hu) an infected WF stinging a healthy human has a certain chance to infect the human. A non-infected WF that stings a contagious human will become infected. An infected human will stay contagious for a certain number of days, after which they stop infecting WF that sting them, and can't become contagious again for the duration of the simulation. For the second case (WF-OS), Eggs layed by infected WF have a small chance to be born infected as well. Table 1 shows the parameters relating to the dengue disease.

A more detailed description of each agent and element can be found in [2].

Table 1. More parameters related to the dengue disease

Parameter	Value
Chance to be infected when stinging an infected human	1
Chance to infect a human	0.9
Chance to be born infected if mother was infected	0.003

5 Tests and Results

To study the propagation of the dengue disease, a number of simulations were performed. The simulations are done in two steps: a) they begin with the initial numbers WM,WF and humans; b) after the system reaches a steady state, a number of infected WF are simultaneously released. The fixed simulation parameters are given in table 2, and the contagious period in humans varies between 8 and 19. For each specific value of the contagious period a total of 40 simulations were run.

Table 2. Model default parameters

Parameter	Value
Model width (blocks)	100
Model height (blocks)	100
Initial number of Male Mosquitoes	1250
Initial number of Female Mosquitoes	750
Initial Number of Humans	700

The results associated with these tests are presented in figures 2 and 3. Figure 2 shows the relation between average infection period in the human population (the amount of the time before no more humans are contagious) and the contagious period. Figure 3 shows the relation between the average number of infected humans and the contagious period.

6 Discussion

Both graphics seem to show an increasing trend in the average infection period and the total number of humans infected with the increase of the contagious period. The correlation of the contagious period with infection period and disease duration follows almost a linear relationship. The fluctuations observed in both figures 2 and 3 are due to the limited number of runs (40); a much higher number would be necessary to reduce the natural variation of the curves due the presence of an accentuated limit cycle.

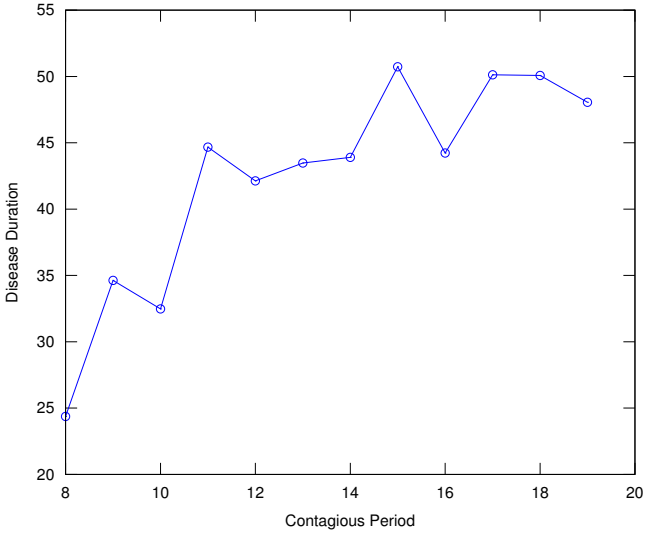


Fig. 2. Relation between the infection period and the contagious period

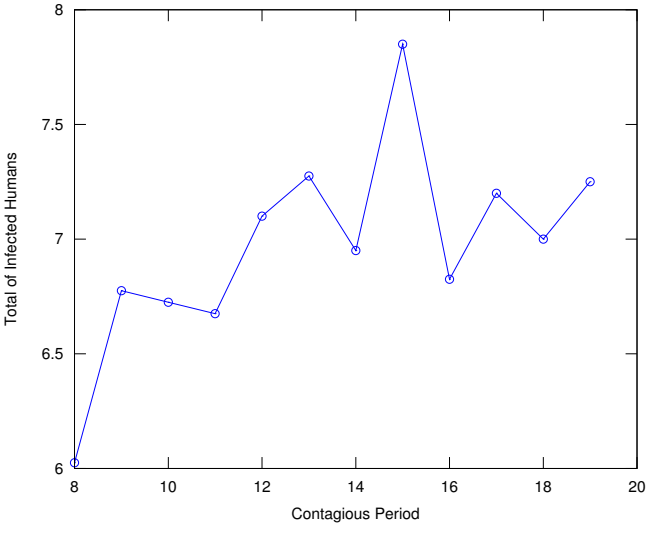


Fig. 3. Relation between the number of humans infected and the contagious period

7 Conclusions and Future Work

The model presented in this paper can be improved by taking into account other important factors which impact mosquito population dynamics, most noticeably environmental factors like temperature, precipitation and wind. The various parameters also need to be fine-tuned to get a model as close to the real population as possible (in particular, some parameters might not be entirely correct in relation to the amount of time associated with each iteration). The various parameters can also be further fine-tuned; in particular, some parameters are not entirely correct in relation to the amount of time associated with each iteration. As such, current results can only be used for a qualitative validation of the model. A more realistic representation of the disease will also be implemented, taking into account the different strains of the dengue virus and immune status of the human population. Simulations studying the impact of population control strategies on the propagation of the dengue disease will also be investigated.

Nevertheless, the current agent based model shows potential as a test bench to help study the propagation of the disease and predict the efficiency of possible treatments before deploying them on the field.

Acknowledgements

This work was partially supported by Fundação para a Ciência e a Tecnologia (ISR/IST plurianual funding) through the POS_Conhecimento Program that includes FEDER funds. The authors C. Isidoro and F. Barata acknowledge their grant BII-2009 to Fundação para a Ciência e Tecnologia (FCT). The author N. Fachada acknowledges its grant SFRH/BD / 48310/2008 to Fundação para a Ciência e Tecnologia (FCT).

References

1. Senior, K.: Climate change and infectious disease: a dangerous liaison. The Lancet Infectious Diseases 8(2), 92–93 (2008)
2. Isidoro, C., Fachada, N., Barata, F., Rosa, A.: Agent-based model of *aedes aegypti* population dynamics. In: Lopes, L.S., Lau, N., Mariano, P., Rocha, L.M. (eds.) EPIA 2009. LNCS (LNAI), vol. 5816, pp. 53–64. Springer, Heidelberg (2009)
3. Helleboogh, A., Vizzari, G., Uhrmacher, A., Michel, F.: Modeling dynamic environments in multi-agent simulation. Autonomous Agents and Multi-Agent Systems 14(1), 87–116 (2007)
4. Ross, R.: The Prevention of Malaria (1911)
5. Macdonald, G.: The analysis of equilibrium in malaria. Trop. Dis. Bull. 49(9), 813–829 (1952)
6. Macdonald, G.: The epidemiology and control of malaria (1957)
7. Eisenberg, J., Reisen, W., Spear, R.: Dynamic model comparing the bionomics of two isolated Culex tarsalis (Diptera: Culicidae) populations: model development. Journal of Medical Entomology 32(2), 83–97 (1995)

8. Eisenberg, J., Reisen, W., Spear, R.: Dynamic model comparing the bionomics of two isolated Culex tarsalis (Diptera: Culicidae) populations: sensitivity analysis. Journal of Medical Entomology 32(2), 98–106 (1995)
9. Alto, B., Juliano, S.: Precipitation and temperature effects on populations of Aedes albopictus (Diptera: Culicidae): implications for range expansion. Journal of Medical Entomology 38(5), 646–656 (2001)
10. Ahumada, J., Lapointe, D., Samuel, M.: Modeling the population dynamics of Culex quinquefasciatus (Diptera: Culicidae), along an elevational gradient in Hawaii. Journal of Medical Entomology 41(6), 1157–1170 (2004)
11. Focks, D., Haile, D., Daniels, E., Mount, G.: Dynamic life table model of a container-inhabiting mosquito, Aedes aegypti (L.)(Diptera: Culicidae). Part 1. Analysis of the literature and model development. Journal of Medical Entomology 30, 1003–1017 (1993)
12. Focks, D., Haile, D., Daniels, E., Mount, G.: Dynamic life table model of a container-inhabiting mosquito, Aedes aegypti (L.)(Diptera: Culicidae). Part 2. Simulation results and validation. Journal of Medical Entomology 30, 1018–1028 (1993)
13. Deng, C., Tao, H., Ye, Z.: Agent-based modeling to simulate the dengue spread 7143, 714310 (2008)
14. Thomas, D., Donnelly, C., Wood, R., Alphey, L.: Insect population control using a dominant, repressible, lethal genetic system. Science 287 (5462), 2474–2476
15. Atkinson, M., Su, Z., Alphey, N., Alphey, L., Coleman, P., Wein, L.: Analyzing the control of mosquito-borne diseases by a dominant lethal genetic system. Proceedings of the National Academy of Sciences 104(22), 9540–9546 (2007)
16. Esteva, L., Mo Yang, H.: Mathematical model to assess the control of Aedes aegypti mosquitoes by the sterile insect technique. Mathematical Biosciences 198(2), 132–147 (2005)
17. Li, J.: Simple mathematical models for interacting wild and transgenic mosquito populations. Mathematical Biosciences 189(1), 39–59 (2004)
18. Maiti, A., Patra, B., Samanta, G.: Sterile insect release method as a control measure of insect pests: A mathematical model. Journal of Applied Mathematics and Computing 22(3), 71–86 (2006)
19. Fachada, N.: Agent-based Simulation of the Immune System. Master's thesis, Instituto Superior Técnico, Lisboa (July 2008)

All in the Same Boat: A "Situated" Model of Emergent Immune Response

Tom Hebbron, Jason Noble, and Seth Bullock

University of Southampton, SO17 1BJ, UK

Abstract. Immune systems provide a unique window on the evolution of individuality. Existing models of immune systems fail to consider them as situated within a biochemical context. We present a model that uses an NK landscape as an underlying metabolic substrate, represents organisms as having both internal and external structure, and provides a basis for studying the coevolution of pathogens and host immune responses. Early results from the model are discussed; we show that interaction between organisms drives a population to optima distinct from those found when adapting against an abiotic background.

1 Immune Systems

From our viewpoint as highly adapted biological agents ourselves, it may seem obvious that there is a boundary between any one individual, its environment, and other individuals in the population. However, if we look at the earliest forms of life, and at the major transitions in evolution, we find that the boundaries of what we would call an individual are not given *a priori*, but are themselves subject to evolutionary change. Furthermore, any form of life, from the simplest autocatalytic set through to a complex multi-cellular animal, represents a local concentration of resources in the environment that could be used by other neighbouring organisms. As the interests of the selfish genes in any opportunistic neighbour are rarely compatible with one's own, a wide variety of mechanisms have arisen to counter such exploitation [1]. The most elaborate of these mechanisms, and a characteristic marker of individuality, is the sophisticated defence of the self/non-self divide that we see in the adaptive immune systems of jawed vertebrates. In everyday life, the adaptive immune system is what makes vaccination possible, and explains why there are many diseases like chickenpox that you are unlikely to catch twice. Preceding adaptive immunity, innate immune mechanisms are a universal feature of even unicellular life [1]. The genome reflects the evolutionary history of pathogenic exposure and successful responses. The continuing selective advantage of these germline-encoded responses depends on the conservation of molecular features of pathogens; features crucial to their metabolism or membrane and difficult to mutate to non-recognised alternatives.

We believe that the study of the immune system is important because it provides a unique window on (and is perhaps even synonymous with) the evolution of individuality itself. Research in immunology is revealing more of the specific

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 353–360, 2011.

mechanisms by which modern immune systems operate, but there is a gap in our understanding of the evolutionary history and adaptive landscape of immune responses. Artificial life provides the modelling tools to fill that gap.

Clearly, exploitation by pathogens creates selection pressure for a defensive response. But some organisms get by with much simpler immune responses than others. By simulating very simple ecologies we hope to discover the necessary and sufficient conditions for the emergence of an immune response, and in further work, more sophisticated adaptive immune systems. Evolutionary simulation also allows us to probe the consequences an immune system has on the subsequent adaptive landscape for hosts and pathogens.

Artificial immune systems (AIS) research is a growing field. In the same vein as earlier work on genetic algorithms and neural networks, researchers have noted that immune systems, and in particular the adaptive immune system, appear to have computationally useful properties (see [2] for a critical review). Computer security is the most popular application area: detecting computer viruses or spam emails for example.

Taking evolved solutions from nature and applying them to human problems is increasingly prevalent in many branches of engineering. However, when borrowing ideas from biology in this way, there can be pitfalls in both under- and over-reaching. For example, an AIS performing binary classification (e.g., "safe" / "dangerous") is not doing what a real immune system does. The algorithm may do well at the practical task we set for it, but it is not embedded in biochemistry in the way a real immune system is. There is a superficial similarity but we would be wrong to draw conclusions about immunobiology from the AIS. Conversely, building a high-definition model of immune system components might be a valuable scientific goal, but as engineering it would be a misplaced effort if the complex contextual constraints of the real immune system were not present in the target problem. It is important to look closely at the differences between the practical problems tackled by AIS researchers and the biological problem faced by real immune systems, as a false assumption of congruence will lead to either models that are too simple to be good science, or tools that are too byzantine to be good engineering.

Immunology to date has been primarily a medical science, concerned with questions of mechanism and ontogeny [3]. In other words, how does the immune system work, how might we fix it when it's broken, and what is its normal course of development? AIS research draws heavily from this work on proximate mechanism. The question of function however, is often left implicit — immune systems are *for* protecting against pathogens. This question warrants further exploration: for there to be pressure for an immune system in the first place requires certain ecological dynamics to hold: pathogenic behaviour must exist, and defences against it must be evolutionarily accessible. Once an immune system emerges it makes fundamental changes to the future evolutionary pathways available to a lineage; some adjacent possible phenotypes will be incompatible with the defenses now encoded in the genome. Simulation modelling is the logical tool for this approach: "For the past century immunology has, with great

success, been occupied with analyzing the immune system into its molecular building-blocks. The field is now ripe for synthesis." [4, p. 30].

Why should we be interested in modelling the function or adaptive value of immune responses? High-resolution, predictive models of biological mechanisms are certainly useful, for instance in developing therapeutics; but such models are opaque. We can see how they work, but not why they work. Such models are also open to misinterpretation: for instance, low iron levels or a fever may not be a symptom of infection but an immune response, creating an unfavourable environment for a pathogen. Without knowing the function of this mechanism, there is the danger that therapeutics are employed to lower the temperature or restore iron levels, working against the biological mechanisms [5]. This need for functional explanations to understand a mechanism in context is also an example of the "no free lunch" theorem [6]: in short, you cannot evaluate an algorithm for solving a problem without some knowledge of the problem and how it arose, and there are no universal solutions — the success of an algorithm on one problem does not necessarily translate to another. (Compare the notion in [7] that evolution produces *nichiversal* and not universal solutions.) This is significant for AIS practitioners, who are trying to isolate the design principles of a system particularly tightly embedded in its biological context.

Parasitic and subsequent coevolutionary behaviour has been observed in artificial life systems before, notably in Tierra [8], which was not explicitly designed to exhibit the phenomenon. Parasitism in Tierra is an interesting example for us because it shows that a fixed and quite brittle substrate defined by Tierra's virtual machine instruction set gave rise to parasitism and the beginnings of an immune response, suggesting that a more general model should find regions of biological interest without much difficulty.

To investigate functional questions about the evolution of the immune system, we must consider the major transitions in its history. We see these as, first, the isolation of metabolism from random perturbations through some form of membrane. Second, once proto-cellular individuals are present, predation will naturally follow as a strategy; i.e., exploiting other organisms as local concentrations of resources and the corresponding coevolution of defensive counter strategies. Third, the accumulation in the genome of these defences constitutes the emergence of innate immunity. (A fourth stage, not represented in the current model, is the emergence of somatic mutations, i.e., adaptive immunity).

2 Constructing a Model of Immune Function

A protective membrane will be assumed in our model. Other ALife researchers have modelled the low-level physicochemical interactions required for the emergence of membranes and proto-cells [9], but in our case starting from an artificial chemistry would make the observation of higher-level strategic events difficult. We abstract the underlying physicochemistry to a set of "metabolic components" and their interactions. An organism is a genetically specified phenotype built from these components, which interact to provide energy payoffs.

Interaction matrix (interval = 3.6)

0.2	0	0	0.6
-1.3	0.1	0	0
0	0	0.5	0.4
-0.1	0	0	0.4

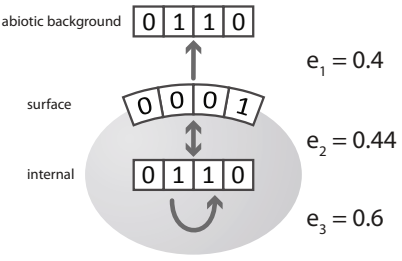


Fig. 1. An example interaction matrix (left) and the calculation of an organism's net energy gain (right). For details see text.

The degree of complexity of the interactions between components is a key parameter of the model; to tune this complexity we borrowed from Kauffman's NK model of tunably rugged fitness landscapes [10]. An N by N interaction matrix defines the energy gain or loss $[-1.0, 1.0]$ of all possible interactions between N metabolic components. Concentration is not modelled, so a metabolic 'state' in the model is represented simply by a bitstring of length N, i.e., a list specifying which components are present. The values in the interaction matrix specify the energy payoff for the column component against the background of the row component (see Figure 1). For example, the upper-right corner of the example matrix indicates that component 4 against a background of component 1 would result in an energy payoff of 0.6. Note that the matrix is not symmetrical, however. A function *payoff(a,b)* gives the energy payoff for state a against a background of state b. This is the sum of all the relevant component-by-component payoffs.

The values in the interaction matrix determine the ruggedness of the fitness landscape that a population of organisms has to search. For example, a simple "Mt Fuji" single-optima landscape can be defined by having only a single row j populated with non-zero energy payoff values. We expect that interesting regions of parameter space will be those with a slightly rugged landscape, not too simple, but not random. Interaction matrices for this type of landscape are generated using an NK-esque algorithm: each row of the matrix is filled with K values drawn from a normal distribution with mean 0.

The evolution of organisms in our model does not take place in a vacuum: a constant abiotic background solution is universally available. Made of the same components as organisms, it is represented by a state bitstring randomly chosen at initialisation, normally with K components present. Organisms themselves are represented by two state bitstrings (each of length N): the first encodes components present in the organism's internal structure, and the second encodes those present on the organism's external surface. This compartmentalisation protects the internal metabolism from outside perturbation and allows organisms to make use of highly reactive components in a controlled environment, whilst exposing selected components well adapted to the rewards and threats of the external environment. Complete decoupling is prevented by requiring that the two sets of components complement one another in order to transfer energy across the

membrane, into the organism for use towards reproduction. The energy chain is represented in the model by three values:

e_1: energy gained or lost in relation to the current external environment: a combination of the abiotic nutrient background bitstring and the surface bitstring of any currently interacting organisms. e_1 = *payoff(surface, context)*.

e_2: a measure of transmission efficiency across the membrane. Calculated as *payoff(surface, internal)* + *payoff(internal, surface)*, normalised using the interval of all matrix values to $[0.0, 1.0]$.

e_3: a measure of the efficiency of the organism's internal metabolism, calculated as *payoff(internal, internal)* + *payoff(internal, internal)*. This evaluates the internal components in the context of themselves.

These three factors are combined to give a total change in energy per timestep of $e_1 \times e_2 \times e_3$, except that if e_1 is negative, the change in energy is equal to e_1 alone, i.e., the organism suffers a loss in energy that is not modulated by its internal metabolism. If an organism's energy level falls below zero, it dies.

Organisms are asexual and reproduce when their stored energy reaches a threshold, which is dependent on the number of components present in their phenotype. In other words, it costs more for larger organisms to reproduce. If we define $onecount(string)$ as the number of ones in a bitstring, then the value of the reproduction threshold is $N + (10 \times (onecount(internal) + onecount(surface))$. At reproduction, the current energy level is split equally between mother and daughter. Mutation of the daughter organism occurs through random flipping of bits in the surface and internal structure strings. A population is initialised by generating a random genotype and initial energy value $[0, N]$ for each organism. During simulation, at each time step the population is shuffled into a random order, this ordering is treated as a ring, and each organism interacts with its left and right neighbours. Energy gains and losses are updated, any offspring are added to the population, dead individuals are removed, and, if necessary, the population is pruned to a predefined carrying capacity by random reaping.

Given a rich interaction matrix and organisms that can interact with each other, the fitness implications of different phenotypes will be complex. In the runs described below, we wanted to limit this complexity by constraining organisms to interact initially only with the nutrient background. This reflects an early ecological stage in which a low-density population experiences no intraspecific competition for resources. In the experiments we will describe here, only a single population is used. However, the model is designed for extension to multiple populations to implement, for example, explicit coevolution between a population of hosts and a population of pathogens.

We anticipate that the primary problem for the evolving organisms in this model will be adaptation to the interaction matrix. This will involve finding a surface structure that works well with the nutrient background, and an internal structure that performs two roles: complementing the surface to pass energy across the membrane, and providing a good internal metabolism by functioning

efficiently in the context of itself. Adapting to the interaction matrix is complicated by the fact that the organisms can interact with each other. A surface structure that works well with the nutrient background may be vulnerable to exploitation by other organisms. Therefore we also expect to see the beginnings of a coevolutionary arms race; exploiting other organisms may give higher payoffs than interacting with the background alone, but when this starts to happen it also provides selection pressure for defensive surface structures, i.e., the beginnings of an immune response.

3 Initial Results and Discussion

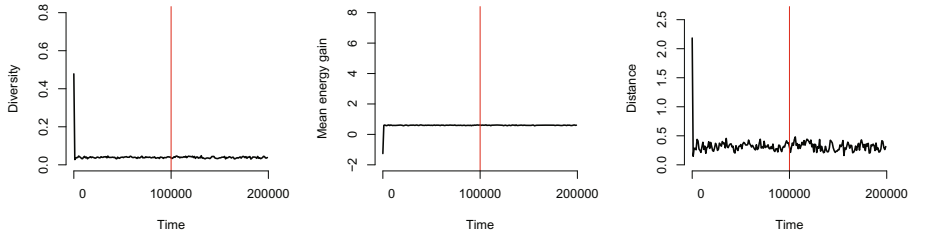
We used a population size of 1000, a mutation rate of 0.02 per locus, and a substrate with $N = 10$ components. The simulation was run for 100,000 timesteps allowing only interaction with the nutrient background, then for a further 100,000 timesteps with inter-organism interactions enabled. Multiple runs were performed, but only a single representative run is described as our aim here is to convey the qualitative dynamics of the model.

We collected data every 200 timesteps on the currently modal genotype, the genetic diversity of the population (measured as the mean disagreement with the modal genotype), the distance between the population centroid and the optimum reached at the end of the initial 100,000 timesteps, and the mean energy gain per timestep across the population.

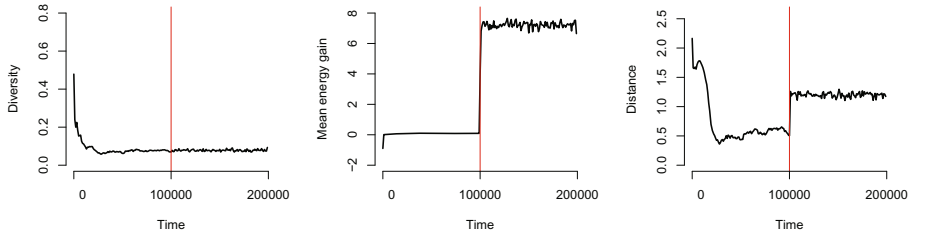
To confirm that the model was functioning as intended, we devised a "Mt. Fuji" interaction matrix whereby only a single row j had non-zero values, with a minimum of 2 positive values. We paired this with a nutrient background where only component j was present. This meant that there was a global optimum genotype with component j present either on the surface or internally (to enable e_2 to be non-zero and energy to be transferred across the membrane) and any components with a positive interaction with j present both internally and on the surface. Figure 2 (top row a-c) shows the results for this condition.

Evolution given the Mt. Fuji interaction matrix confirmed our expectations. Many organisms in the initial population died off immediately as they had negative net energy gain. The few viable individuals rapidly converged on the anticipated global optimum. Enabling inter-organism interactions at $t = 100,000$ had no effect, but this was also expected as the nature of the payoff matrix meant that there were no benefits available from interaction that were not already available against the background.

Having established that the simulation was producing intelligible results, we constructed a richer interaction matrix to serve as an initial exploration of the space of possible substrates. As described in section 2, an NK-like interaction matrix was constructed with $K = 2$ and payoffs drawn from a normal distribution with a mean of zero and a standard deviation of 0.5. The corresponding nutrient background had two randomly selected components present. In this case, the landscape was more rugged so we expected greater difficulty for the population in finding an initial optimum. We also expected that once



(a) genetic diversity (b) mean energy gain (c) target distance



(d) genetic diversity (e) mean energy gain (f) target distance

Fig. 2. Population genetic diversity, mean per-timestep energy gain, and distance of population centroid from a target optimum over time. Graphs a-c show the Mt. Fuji case; graphs d-f show the rich matrix case. Vertical line at time 100,000 indicates the switching on of interaction.

inter-organism interactions were enabled, there would be a significant change in mean energy gain as many more interaction possibilities became available.

Results for this "rich matrix" condition are shown in Figure 2 (bottom row d-f). Compared to the "Mt. Fuji" condition, genetic diversity remains higher for longer, as the initial no-interaction population searches the rugged landscape and discovers multiple local optima. As the individuals on the fittest of these optima begin to dominate the population, diversity falls: Figure 2f confirms that there is significant movement in the population centroid over this period. Figure 2e shows that mean energy gain, on the other hand, increases only gradually. Once interaction is enabled at $t = 100,000$, new opportunities to exploit the surfaces of other organisms for energy gain become available. These translate as new and higher optima. Genetic diversity is not appreciably affected, but mean energy gain sees a large increase, and Figure 2f shows that the population rapidly converge on the new optima.

At $t = 100,000$ the modal genotype was `0011101100-0011111100` (internal-surface). At the end of the run ($t = 200,000$), and after the effects of interaction, the modal genotype was `0011101100-0011101000`. Note that the internal structure of the modal organism has not changed, but two surface components have been lost. If we examine the $e_1 \times e_2 \times e_3$ energy process for this final modal

organism, we find that although it is well adapted to the interactive context, it in fact represents a phenotype that would not have been viable in the earlier part of the run, i.e., its energy input in the context of the nutrient background, e_1, would have been zero. Note also that the surface and internal components are almost identical, differing only in one position. This makes sense: a state that works well with itself, as an internal structure, gives you a high e_3 value. If this structure is also your surface, then similarly e_2 will be high. Finally, by the same logic, if this string is also the modal surface structure, high values of e_1 will be gained on all sides when organisms interact with each other.

Thus it seems we have discovered a commensal evolutionarily stable state for a single population, rather than having demonstrated the kind of coevolutionary arms race we ultimately would like to see in the model. However, this is not entirely surprising, as coevolution is extremely difficult to engineer in a single population context. In the short term, the obvious next step for the model will be to achieve this by using multiple weakly interacting populations to avoid premature convergence. It may be necessary to explicitly define hosts and pathogens rather than wait for them to evolve; e.g., a host population with a low mutation rate and a minimum number of components per organism (hosts must be large) and a pathogen population with the opposite properties (fast-evolving and small). Beyond that, further development of the model framework will allow us to consider multi-cellular organisms, danger signals [11], and ultimately adaptive immune systems.

Acknowledgements

This work was partly supported by EPSRC grant: EP/D00232X/1.

References

1. Danilova, N.: The evolution of immune mechanisms. J. Exp. Zool. Part B 306B(6), 496–520 (2006)
2. Garrett, S.M.: How Do We Evaluate Artificial Immune Systems? Evol. Comput. 13(2), 145–177 (2005)
3. Murphy, K.: Janeway's Immunobiology. Garland Science (2007)
4. Cohen, I.R.: Tending Adam's Garden. Academic Press, London (2004)
5. Nesse, R.M., Williams, G.C.: Why We Get Sick. Vintage (1996)
6. Wolpert, D.H., Macready, W.G.: No free lunch theorems for search. IEEE. T. Evolut. Comput. 1(1), 67–82 (1997)
7. Bullock, S.: The fallacy of general purpose bio-inspired computing. In: Rocha, L.M., et al. (eds.) Tenth Intl. Conf. on Artificial Life, pp. 540–545. MIT, Cambridge (2006)
8. Ray, T.S., Xu, C.: Measures of evolvability in Tierra. Artif. Life and Robotics 5(4), 211–214 (2001)
9. McMullin, B.: Thirty years of computational autopoiesis: a review. Artif. Life 10(3), 277–295 (2004)
10. Kauffman, S.A.: The Origins of Order. OUP (1993)
11. Matzinger, P.: The danger model: a renewed sense of self. Science 296(5566), 301–305 (2002)

Multi-Agent Model for Simulation at the Subcellular Level

M. Beurton-Aimar, N. Parisey, and F. Vallée

Laboratoire Bordelais de de Recherche en Informatique - UMR CNRS 5800 -
University Bordeaux 1 et 2
(beurton,parisey,vallee)@labri.fr

Abstract. From the beginning of biological modeling, simulations were an efficient way to understand local mechanisms linked to whole system behaviors. Cellular automata and more recently Multi-Agent Systems (MAS) are currently used to model ecological systems. Virus dissemination through population or insect collaborations are well known examples of how simple interactions between entities (agents) are able to build a complex situation at the level of the whole population. But use of MAS to design biological system at the cellular or subcellular level, mainly for enzymatic reactions, is a relatively new application of the agent paradigm. In fact, agents used for ecology simulations are 'non-physical' agents, i.e. in general they do not have any explicit representation of their geometry, their space bulk or the articulated movements of their body parts. These characteristics are essential in enzymatic behaviors. The three dimension structure and movements of this structure condition the realisation of the reaction that can be stopped or conversely favored by specific conformations. In order to simulate subcellular biological processes, we defined agents capable of simulating molecular conformational changes. These agents integrate molecular modeling data, for conformational change methods, and biological ontology data, for conformational change conditions. As some biological entities are motor of the enzymatic reactions while others are simple partners, we defined two agent subtypes, active and passive agents. As a proof of concept, we applied our model to the simulation of enzymatic oxydo-reduction reactions.

Introduction

Metabolic processes are mainly biochemical processes. They work at the subcellular level. In general, for one species they implicate a few hundreds of molecule types. The complexity of such processes is not due to the complexity of each mechanism of metabolic reactions but more to the multiplicity of used molecules and competitiveness between reactions to capture and to transform molecules. The whole set of reactions implicated in a specific metabolism is a network which can be described as an interaction graph where the nodes are metabolites and the edges are enzymes which catalyze reactions. Understanding how the modification of one reaction process influences other ones through the network is a key point

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 361–368, 2011.

in system biology research. The chains of metabolic reactions have been currently modelled by reaction-diffusion equations [1] and simulated by differential equations in general [2]. But these models ignore a part of metabolic processes characteristics like those linked to the spatial molecule structures. They do not also take into account information about the molecule location. These information can be very important if there is a kind of competition between processes to use some molecules.

Multi-agent systems are well known to simulate simple local behavior of entities which generate complex behavior at the system scale[3]. Many examples exist where agent modelling has been able to provide realistic explanations of what happens in *"real life"*. Most famous ones are the way to find the shortest path to food by ants or the capability for some birds to keep a specific organisation during their flight[4]. In these models the agents are described as *reactive* agents. Complex agent models have been made to mimic the behavior of human societies or to test hypotheses in economy or in artificial intelligence. These models most often use *cognitive* agents, i.e. agents able to plan their action, to build strategy and to take decisions. At the cellular or subcellular level it is not appropriated to talk of molecule *intentions*, so if molecules are agents it is a kind of evidence that they are reactive agents. One can note that most often all these models do not ever propose to represent explicitly the geometry or the physical structures of agents. In our project BASiL to model metabolism of species with a multi-agent system, we have defined an extended reactive agent model able to take into account physical properties of implicated molecules, their internal movements and their interaction at different levels. After a presentation of the type of processes we need to model, we will explain the BASiL agent model and we will finish with a concrete example of the modelling of the electron transfer chain into the mitochondrial respiratory chain.

1 Biological Processes at the Subcellular Level

Metabolic processes imply proteins or enzymatic complexes, and metabolites which interact at the intra or inter molecular levels. Most often, metabolites are small molecules and enzymes macromolecules. Enzymes exhibit capabilities to capture molecules through binding sites and each reaction they catalyse is characterized by a reaction velocity. Enzymatic reactions can be influenced by several elements: presence or absence of inhibitors/activators, environment parameters like temperature and so on. All these elements affect both external and internal macromolecular motion. For example temperature augmentation increase the velocity of the enzyme in a medium (for example, a cytoplasm). We should also note that enzymes are composed of a huge number of atoms and that temperature can also affect the relative spatial arrangement of their atoms (as studied in stereochemistry). This means that a macromolecule can assume different shapes. It is well known in biology that the probability of an enzymatic reaction to occur is controlled by the macromolecule current shape [5] [6].

Hence, in order to study the enzymatic reactions of a system it is necessary to assess the molecules internal movement and their interactions with other molecules. This fact gave rise to the field of Molecular Dynamics (MD)[7] which study the dynamical nature of molecular motions. Most MD algorithms have a high computational cost and they can simulate, at most, nanoseconds of conformational changes. Unfortunately, enzymatic reactions can occur between microseconds and milliseconds. Hence, we have apparently incompatible time scales.

As it is not possible to simulate movements at the atom level in the same time scale than the biochemical reactions, we have used algorithms able to compute movements at the molecular level. We use geometrically based algorithm [8] to transform a macromolecule into a set of Constrained Rigid Bodies (CRB). With such an algorithm, we can resume a molecule as a set of small parts linked by hinges and able to move around axes. This enable us to accurately describe internal movements of a macromolecule. Interactions at the molecular level can be resumed as interactions between parts of each molecule. Models coming from molecular dynamics provide tools like "Coarse graining approach"[9] to compute attraction/repulsion between parts, grains, of macromolecules. By using the previously predicted internal parts of a macromolecule, we can convert set of physical laws at the atomic level to forces fields resumed on grains.

2 Multi-Agent Model for Macromolecules

Traditionally, multi-agent model defined two types of agents: *cognitive* agents and *reactive* agents. The main differences between them is that *cognitive* agents are able to apply strategy to take decisions about their behavior. They can choose by their own what they *"want"* to do. *Reactive* agents only communicate through their environment and just react from a kind of stimulus. It is difficult to claim that molecules take decisions about *"what they do"*, so it is a natural way to choose *reactive* agents to design MAS for molecular processes. Our model is based on such agent but we have completed the model by adding a `Generic-body` attribute to represent all the useful characteristics.

We have developed BASiL, a framework dedicated to simulations at the cellular and subcellular level. Its architecture contains a generic class `Agent` which is the main container to design a reactive agent. Its attributes are: `Description` and `Generic-body`. A `Description` is a class where we store the information relevant to the application domain but not directly useful for the simulations. At present, we store biological data like the biological name of the molecules, the references to bibliography and details given by biological databases in subclasses of `Description`. The organisation of these subclasses comes from BioPAX. BioPAX [10] is a formalism dedicated to the description of biological objects and specifically to molecules at the subcellular level. These subclasses are stored into the `biological-model` package. Of course, the `Description` class can be sub-classed by any another class able to give the agent domain description.

The `Generic-body` class is the place to design the physical properties of the agents. The three classes: `Agent`,`Description` and `Generic-body` are the core of the generic `agent-model` package.

2.1 Modelling Molecule Body

One of the possible implementations of the molecule physical properties is to use information given by molecular dynamics. We have defined a set of classes to describe a multiple-grain model for the agent body. The class `Grain-body` instantiates the agent body as a generic grain which can be a set of grains (following the composite design patterns [11]). The grain implements a set of forces defining what happens at this level when a grain interacts another one.

2.2 Modelling the Environment

Agents are situated in an environment. The way to implement this environment conditions strongly the algorithm complexity when we implement the life cycle of each agent. Indeed, one of the main actions of agents at each time step is to observe their neighborhood and to fire several interactions with agents belong to it. The space is divided into small parts organized as grid or any structured set. Grid is efficient if the population of agents is big enough. Tree structures can be used if some places are empty and if it is not adequate to visit the whole space to find agents.

We have chosen to implement firstly the environment as a grid. To keep the capability to change that and to switch easily to a tree structure or any another one, we have designed the `Spatial-Env` as a generic class with concrete subclasses like `Grid`. Each box in the grid contains a list of agents situated into this box. So agents only know their relative position into the box and access to their neighborhood through the neighboring boxes.

2.3 Modelling Enzymatic Processes

As it is mentioned previously, metabolism reactions are mainly biochemical reactions. Biochemical reactions imply enzymes as catalyst and metabolites as substrates or products. For example, we can describe enzymatic reactions as follow:

$M_a + M_b = M_c$ if they meet E_1

$M_c + M_a = M_d$ if they meet E_2

`Meet` means that the substrates M_a and M_b are able to be captured by E_1 (through E_1 binding sites) and that they are chemically transformed into M_c.

Agents hold a set of actions to perform at each simulation time step. As metabolites are not able to capture other agents, they have only the capability to diffuse through the environment. At the opposite, enzymes hold the scheme of the biochemical reactions they catalyze. Therefore, we have designed two subclasses of agents:

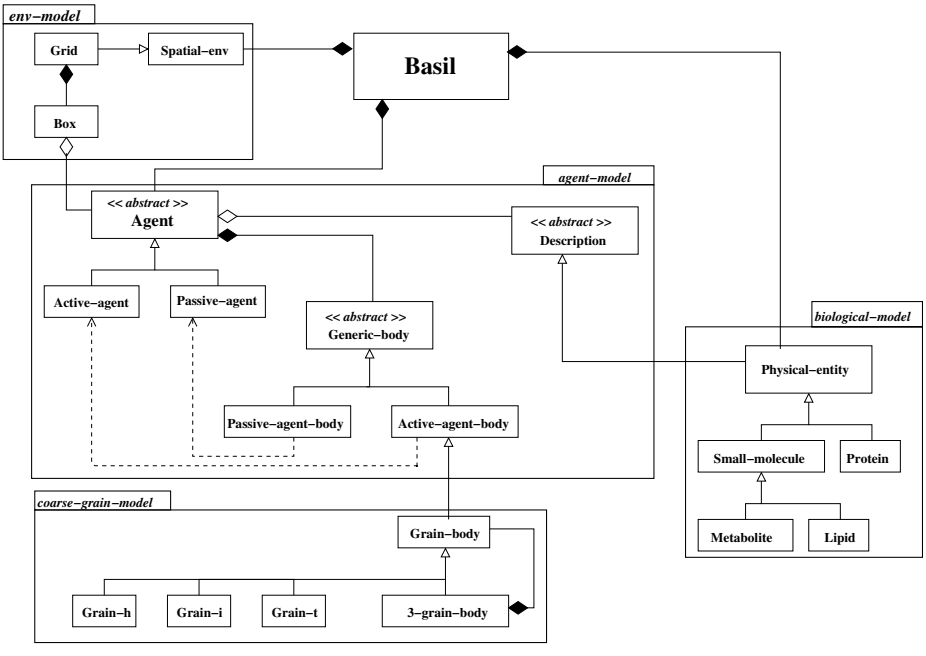


Fig. 1. BASiL architecture

– one to represent agents having initiative for starting interactions. This is the `Active agent` class.
– another one to represent agents undergoing Brownian motions without any more action. This is the `Passive agent` class.

Coding metabolites as `passive agents` allows us to save a lot of computational time because the quantity of metabolites is more than hundred times higher than the quantity of enzymes and `passive agents` do not need to observe their neighborhood. Figure 1 shows the main components of the BASiL architecture.

3 Simulation of Enzymatic Reactions

BASiL framework have been used to simulate electron transfer through Respiratory Chain (RC) complexes [12]. We will now describe the work that we have done about the Complex III of the RC. From the PDB files [13] we have run the algorithm to find the Complex III rigid parts. Figure 2 shows the results of this computing procedures and where are the binding sites involved in the electron transfer function. Each three parts can be considered as grain. Each grain stores a set of force fields based on Lennard-Jones potential [9]. That enables to simulate the conformational changes the molecule can exhibit.

The electron transfer rate between two redox elements clustered in a macromolecule has been described by Dutton and Moser [14]. They defined **a set of**

phenomenological equations that links the transfer rate to the distance (in Angstrom) between redox elements. The following two equations apply in case of endergonic (ke_{end}) and exergonic (ke_{ex}) reactions:

$$log_{10}(ke_{end}) = 15 - 0.6 * R - 3.1 * \frac{(\Delta G^0 + \lambda)^2}{\lambda} \quad (1)$$

$$log_{10}(ke_{ex}) = 15 - 0.6 * R - 3.1 * \frac{(-\Delta G^0 + \lambda)^2}{\lambda} - \frac{\Delta G^0}{0.06} \quad (2)$$

where:

- ke is the transfer rate in electron(s) per second,
- R is the Euclidean distance between two redox elements,
- ΔG^0 is the free energy of the redox system,
- and λ is the reorganization energy of the system.

As R changes more frequently than ΔG^0. Therefore, in our method, we consider ΔG^0 constant over a time step in order to examine the influence of R changes. At each time step of our simulations, we used equations 1 and 2 to compute probabilities of electron transfer between two redox elements.

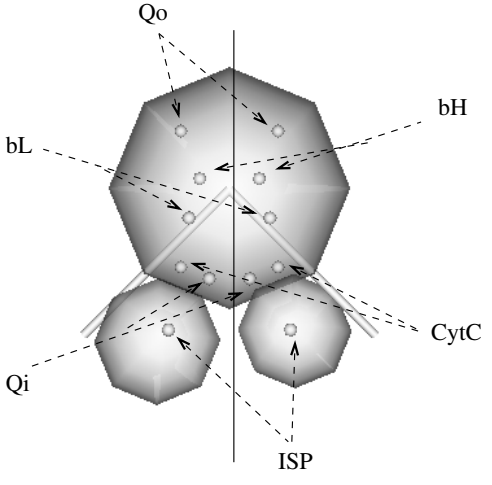


Fig. 2. Complex III partitioning with its 6 binding sites per monomer

3.1 Simulation Results

At first, by simply tracking sites occupation over time, we were able to differentiate between situations leading to similar kinetics but being produced by different electron paths. These early results lead us to define metrics for Complex III "efficiency". Complex III **efficiency** is usually defined as its ability to **generate bifurcated electron transfer** from quinone clustered in the Qo site to cytochrome c1 and quinone clustered in the Qi site (cf fig. 2). Hence, we decided over a few parameters to assess efficiency in our simulations:

– time latency for a redox reaction to occur after `Passive Agent` binding,
– electron transfer background noise, i.e. percentage of electron exchange where an electron goes back to where it started,
– percentage of Qi reaction over global redox reactions.

Using those metrics, we looked more closely into our simulation trace logs and found that a first Qi reaction peak is linked with a majority of canonical Q cycle while the second peak is due to what was earlier [15] described as a short-circuits, i.e. an electron exchange detrimental to the proton gradient. Taking into account internal dynamics, these two solutions are acceptable as far as the internal macromolecule functions are concerned. Redox potential control short-circuits [2] but, in any case, internal dynamics can affect electron transfer so that this control is broken. Whether, internal dynamics or redox potential is preponderant *in vivo* is a pending question waiting for more data.

4 Conclusion and Future Works

Modelling metabolic processes is a difficult task due to the diversity of molecules implied in biochemical reactions. Using multi-agent system allows us to tackle complexity of such simulations by describing a reactive agent for each molecule. Each agent hold its physical properties into a set of grains which interact together. We have used our model to describe electron transfer in the respiratory chain and we are able to test the influence of the movements of macromolecule on their efficiency. We now plan to add another types of molecules like phospholipids around the macromolecules and to simulate a piece of membrane with its membrane enzymes.

Acknowledgements

We thank the Epigenomic work group of the Evry Genopole and the ANR MitoScoP for different fundings and very helpful scientific discussions.

References

1. Kolmogoroff, A., Pretrovsky, I., Piscounoff, N.: Étude de l'équation de la diffusion avec croissance de la quantite de matière et son application à un problème biologique. Moscow University Mathematics Bulletin (1937)
2. Moser, C.C., Farid, T.A., Chobot, S.E., Dutton, P.L.: Electron tunneling chains of mitochondria. Biochim. Biophys. Acta 1757, 1096–1109 (2006)
3. Ferber, J., Gutknecht, O., Michel, F.: From agents to organizations: An organizational view of multi-agent systems. In: Giorgini, P., Müller, J.P., Odell, J.J. (eds.) AOSE 2003. LNCS, vol. 2935, pp. 214–230. Springer, Heidelberg (2004)
4. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics 21, 25–34 (1987)
5. Monod, J., Wyman, J., Changeux, J.P.: On the nature of allosteric transitions: a plausible model. J. Mol. Biol. 12, 88–118 (1965)

6. Parisey, N., Beurton-Aimar, M.: Investigating oxidoreduction kinetics using protein dynamics. Journal of Biological Physics and Chemistry (2009) (in press)
7. Arkhipov, A., Yin, Y., Schulten, K.: Four-scale description of membrane sculpting by BAR domains. Biophys. J. 95, 2806–2821 (2008)
8. Wriggers, W., Schulten, K.: Protein domain movements: detection of rigid domains and visualization of hinges in comparisons of atomic coordinates. Proteins 29, 1–14 (1997)
9. Marrink, S.J., de Vries, A.H., Tieleman, D.P.: Lipids on the move: simulations of membrane pores, domains, stalks and curves. Biochim. Biophys. Acta 1788, 149–168 (2009)
10. Luciano, J.S.: PAX of mind for pathway researchers. Drug Discov. Today 10, 937–942 (2005)
11. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Addison-Wesley Professional, Reading (1995)
12. Mitchell, P.: The protonmotive Q cycle: a general formulation. FEBS Lett. 59, 137–139 (1975)
13. Bernstein, F.C., Koetzle, T.F., Williams, G.J., Meyer, E.F.J., Brice, M.D., Rodgers, J.R., Kennard, O., Shimanouchi, T., Tasumi, M.: The Protein Data Bank. A computer-based archival file for macromolecular structures. Eur. J. Biochem. 80, 319–324 (1977)
14. Moser, C.C., Keske, J.M., Warncke, K., Farid, R.S., Dutton, P.L.: Nature of biological electron transfer. Nature 355, 796–802 (1992)
15. Osyczka, A., Moser, C.C., Dutton, P.L.: Fixing the Q cycle. Trends Biochem. Sci. 30, 176–182 (2005)

Visualising Random Boolean Network Dynamics: Effects of Perturbations and Canalisation

Susan Stepney

Department of Computer Science, University of York, UK
susan@cs.york.ac.uk

Abstract. We have proposed a simple approach to visualising the time behaviour of Random Boolean Networks (RBNs). Here we demonstrate the approach in a variety of cases: examining the effect of state and structure mutations, and examining the effect of canalising functions for $K > 2$ networks.

1 Introduction

Random Boolean networks (RBNs) are a well-studied form of complex discrete dynamical systems [1,2,3,4,5]. Visualisation of the dynamics can aid understanding, but (unlike for 1D Cellular Automata, for example), there has been no satisfactory visualisation of RBN time behaviour. In [6] we proposed a simple approach to visualising the time behaviour of RBNs; here we demonstrate the approach in a variety of cases: examining the effect of state and structure mutations, and examining the effect of canalising functions for $K > 2$ networks.

2 RBNs

A Random Boolean Network (RBN) comprises N *nodes*. Each node i at time t has a binary valued state, $c_{i,t} \in \mathcal{B}$. Each node has K inputs assigned randomly from K of the N nodes (an input may be from the node itself); the wiring pattern is fixed throughout the lifetime of the network. This wiring defines the node's neighbourhood, $\nu_i \in N^K$.

The state of node i's neighbourhood at time t is $\chi_{i,t} \in \mathcal{B}^K$, a K-tuple of node states that is the projection of the full state onto the neighbourhood ν_i.

Each node has its own randomly chosen local state transition rule, or update rule, $\phi_i : \mathcal{B}^K \rightarrow \mathcal{B}$. These nodes form a network of state transition machines. At each timestep, the state of each node is updated in parallel, $c_{i,t+1} = \phi_i(\chi_{i,t})$.

The global dynamics f is determined by the local rules ϕ_i and the connectivity pattern of the nodes ν_i.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 369–376, 2011.

Kauffman [3,4] investigates the properties of RBNs[1] as a function of connectivity K. Despite all their randomness, "such networks can exhibit powerfully ordered dynamics" [3], particularly when $K = 2$. Kauffman investigates RBNs as simplified models of gene regulatory networks (GRNs). He notes that "cell types are constrained and apparently stable recurrent patterns of gene expression", and interprets his RBN results as demonstrating that a "cell type corresponds to a state cycle attractor" [4, p.467] (in a $K = 2$ network).

Drossel [1] notes that subsequent computer simulation of much larger networks shows that "for larger N the apparent square-root law [of attractor numbers and lengths] does not hold any more, but that the increase with system size is faster".

3 Visualising the Dynamics

Good visualisations can aid the understanding of complex systems, and can help generate new questions and hypotheses about their behaviours.

Kauffman [4, p.203] observes that $K = 2$ RBNs "develop a connected mesh, or *frozen core*, of elements, each frozen in either the 1 or 0 state." We can use this result to provide an order for placing the nodes in the visualisation. Nodes frozen in the 1 or 0 state are placed towards the edges of the figure; nodes that are changing state are placed towards the centre: see figure 1. The different transient behaviours and attractors are clearly visible; for example, it is clear that these show three different attractors, with three different periods.

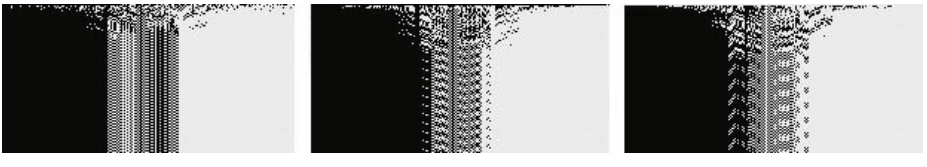


Fig. 1. Visualisation of the time evolution of a $K = 2$ RBN from different initial conditions (half on, half off; all off; all on) with the nodes sorted to expose the frozen core (as described in [6])

A simple algorithm to achieve this node sorting is described in [6]. It results in the frozen core nodes moving to the edges of the figure, whilst the nodes with cycling states are in the centre. Additionally, the frozen core nodes with shorter transient behaviour are closer to the edges than those with longer transient behaviours. Similarly, nodes with cycling states are sorted according to the amount

[1] The wiring conditions given here are not stated explicitly in those references. However, in the $K = N$ case, Kauffman [4, p.192] states that "Since each element receives an input from all other elements, there is only one possible wiring diagram". This implies that multiple connections from a single node are not allowed in a true RBN (otherwise more wiring diagrams would be possible) whereas self connections are allowed (otherwise K would be restricted to a maximum value of $N - 1$). Subsequent definitions (for example [1]) explicitly use the same conditions as given here.

of time they spend in one state or the other, with those half the time in each state towards the centre. This tends to highlight the attractor structure.

Note, however, that the precise order of the nodes depends on the various initial states chosen. In all the examples given here, for simplicity, the network was run only from the all zeroes and from the all ones state to determine the sort order. In figure 1, it can be seen that in the all ones initial state (middle column) the central node is always on, whilst in the all zeroes initial state (right column) it is always off. (This implies it is a node with a self-connection.) Hence, when these two cases are combined, it is on for an average of half the time, and so ends in the centre.

4 Examples

In this section, we explore some different aspects of RBNs, using this visualisation approach to expose the relevant features.

We use Tufte's "small multiples" [7] technique, which "allows the viewer to focus on changes in the data", by displaying an array of RBNs that can be readily compared.

The aim is to use the visualisation to prime intuition and aid understanding of RBNs' rich dynamics, and to provoke hypotheses about the detailed behaviour. Any such hypotheses would need to be investigated in a rigorous manner.

4.1 Perturbing RBN State

Here we visualise the stability of $K = 2$ networks to perturbations of their state.

Kauffman [3] defines a *minimal perturbation* to the state of an RBN as flipping the state of a single node at one timestep. Flipping the state of node i at time t is equivalent to changing its update rule at time $t-1$ to be $c_{i,t} = \neg\phi_i(\chi_{i,t-1})$. Such a perturbation leaves the underlying dynamics, and hence the attractor basin structure, the same, it merely moves the current state to a different position in the state space, from where it continues to evolve under the original dynamics: it is a transient perturbation to the state.

Kauffman [3] describes the *stability* of RBN attractors to minimal perturbations: if the system is on an attractor and suffers a minimal perturbation, does it return to the same attractor, or move to a different one? Is the system *homeostatic*? (Homeostasis is the tendency to maintain a constant state, and to restore its state if perturbed.)

Kauffman [4] describes the *reachability* of other attractors after a minimal perturbation: if the system moves to a different attractor, is it likely to move to any other attractor, or just a subset of them? If the current attractor is considered the analogue of "cell type", how many other types can it *differentiate* into under minimal perturbation?

Kauffman's results pick out the $K = 2$ networks as having interesting behaviour under minimal perturbation (high stability so a perturbation usually has no effect; low reachability so when a perturbation moves the system to another attractor, it moves it to one of only a small subset of possible attractors).

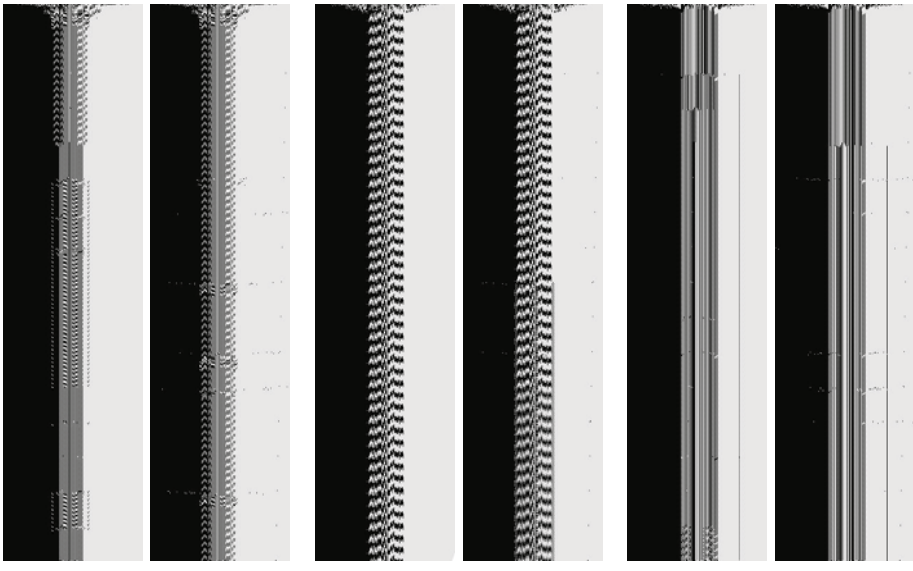


Fig. 2. Visualisation of the time evolution of three typical $K = 2$ RBNs (two runs of each), with $N = 200$, 800 timesteps, and initial condition all nodes "off". After 100 timesteps, a node is flipped once every 50 timesteps. For the left run of each pair, a node is flipped near the centre; for the right run, a node flipped in the frozen core.

Visualisations of the effect of minimal perturbations are shown in figure 2, for perturbations of cycling nodes, and of frozen core nodes.

These visualisations demonstrate that $K = 2$ RBNs are remarkably stable to minimal perturbations. They also suggest further possible properties: (a) a perturbation to a frozen core node is more likely to preserve the attractor than a perturbation to a cycling node; (b) a perturbation to a frozen core node tends to have longer transient behaviour than a perturbation to a cycling node.

4.2 Perturbing RBN Structure

Here we visualise the stability of $K = 2$ networks to perturbations of their structure.

Kauffman [3] defines a *structural perturbation* to an RBN as being a permanent mutation in the connectivity or in the boolean function. So a structural perturbation at time t_0 could change the update rule of node i at all time $t > t_0$ to be ϕ'_i or change the neighbourhood of node i at all time $t > t_0$ to be ν'_i. Since the dynamics is defined by all the ϕ_i and ν_i, such a perturbation changes the underlying dynamics, and hence the attractor basin structure: it is a permanent perturbation to the dynamics, yielding a new RBN.

Such a perturbation could have several consequences: a state previously on an attractor cycle might become a transient state; a state previously on a cycle might move to a cycle of different length, comprising different states; a state might move from an attractor with a small basin of attraction to one with a

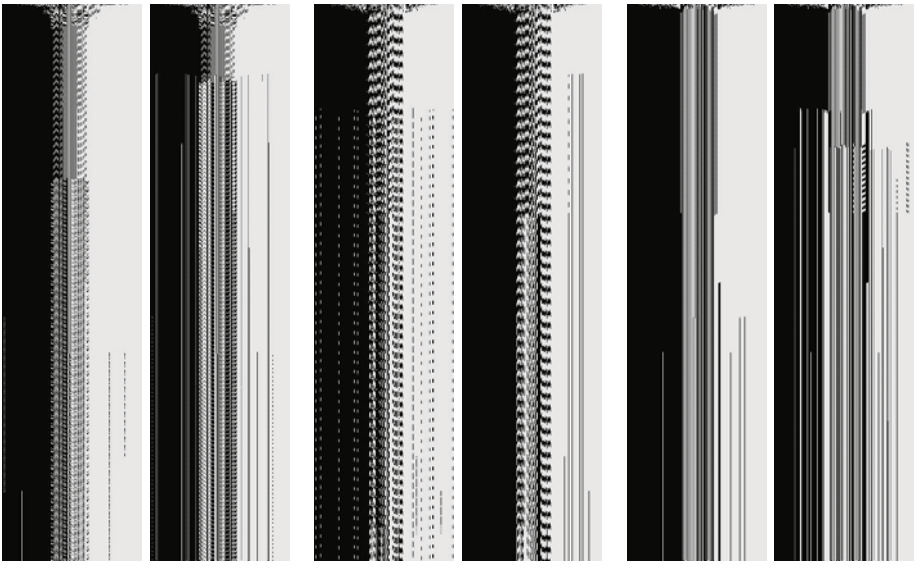


Fig. 3. Visualisation of the time evolution of three typical $K = 2$ RBNs (two runs of each), with $N = 200$, 800 timesteps, and initial condition all nodes "off". After 100 timesteps, the structure of one randomly chosen node is mutated once every 50 timesteps. For the left run of each pair, one of the node's inputs is randomly reassigned; for the right run, the node's boolean function is randomly changed.

large basin; a state might move from a stable (homeostatic) attractor to an unstable attractor; and so on.

Kauffman [4] relates structural perturbation to the *mutation* of a cell; if there is only a small change to the dynamics, this represents mutation to a "similar" kind of cell.

Visualisations of the effect of structural perturbations are shown in figure 3, for perturbations of input connections, and of boolean functions.

These visualisations appear to show that the effect of an input change is less dramatic than that of a boolean function change. Here no distinction is drawn between changing a cycling node or a frozen node: visualisation of further experiments along these lines could yield interesting conjectures about the stability of these RBNs.

4.3 Canalisation

Here we visualise the effect of canalising functions on the time behaviour of $K > 2$ networks.

Kauffman [4, p.203] defines a canalising function as "any Boolean function having the property that it has at least one input having at least one value (1 or 0) which suffices to guarantee that the regulated element assumes a specific value (1 or 0)". ([1] categorises canalising functions further, into weak, strong, and constant). Kauffman argues that the canalising functions are important for

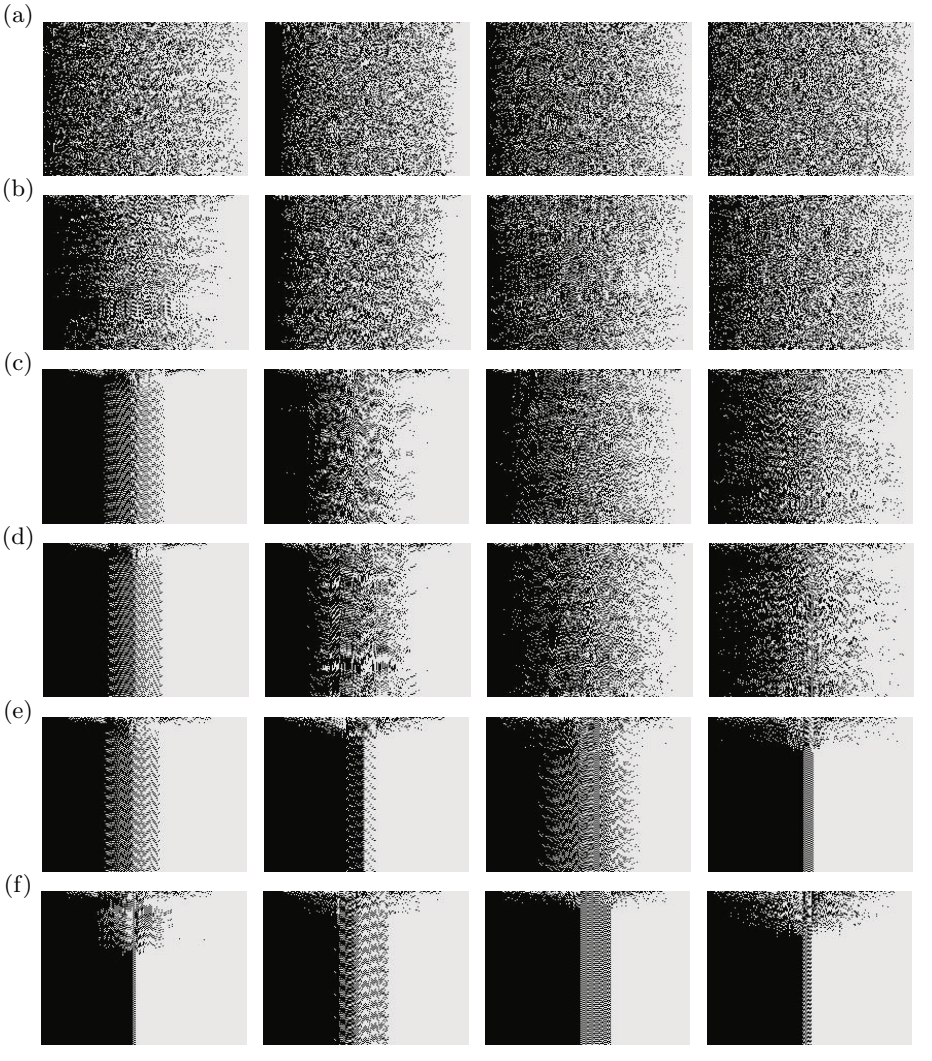


Fig. 4. Visualisation of the time evolution of 24 typical $K = 3$ RBNs, with $N = 200$, and initial condition all nodes "off"; for 150 timesteps; rows have the following number of canalised nodes: (a) 94 = 47.0% (b) 128 = 64.0% (c) 181 = 90.5% (d) 184 = 92.0% (e) 190 = 95.0% (f) 198 = 99.0%

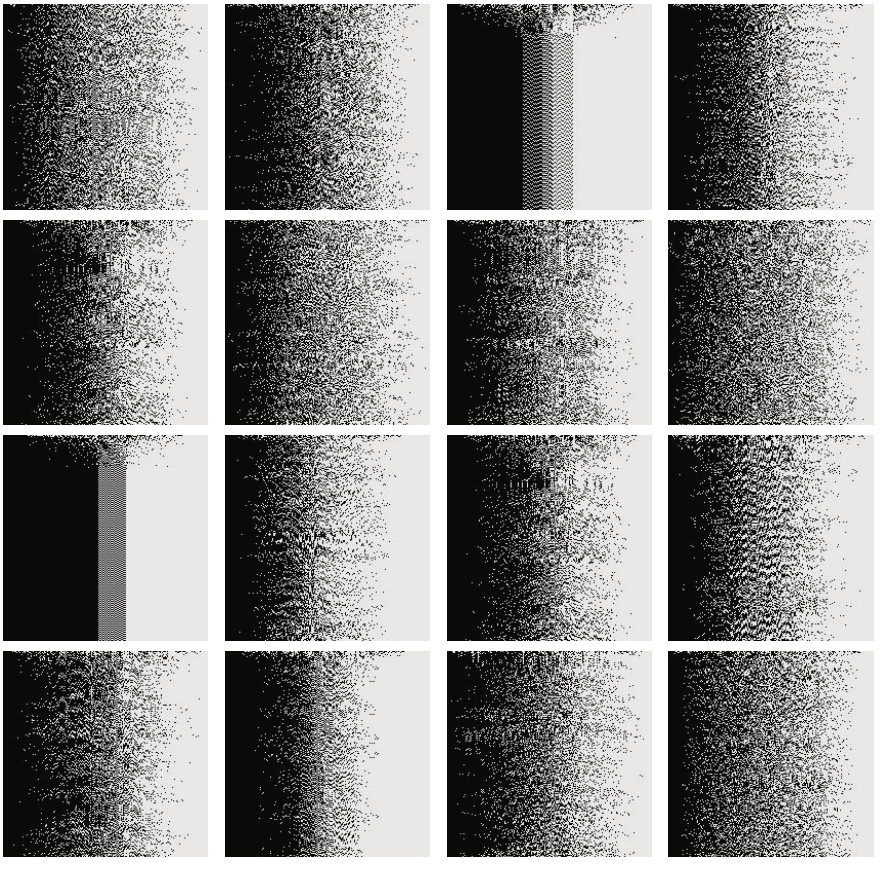


Fig. 5. Visualisation of the time evolution of 16 typical $K = 4$ RBNs, with $N = 200$, and initial condition all nodes "off"; for 200 timesteps; all functions canalising

establishing the frozen core and ordered dynamics of $K = 2$ networks. The proportion of canalising functions decreases rapidly with increasing K . Kauffman [4, p.206] states that "networks with $K > 2$ restricted to canalyzing functions . . . [have] orderly dynamics in the entire network".

Visualisations of the effect of canalising functions on the time behaviour are shown in figures 4 and 5. Clearly for $K = 3$ (figure 4), increasing the proportion of canalising functions does make transients and attractors shorter, and establish an "orderly dynamics". However, for $K = 4$, even with all functions canalising, change in the chaotic behaviour is evident in only a minority of cases. The effect does not appear to be as strong as Kauffman suggests.

5 Discussion and Conclusions

A very simple algorithm allows the time behaviour of RBNs to be visualised in a manner that exposes the transient behaviour, and the structure of the frozen

core and cycling nodes. We have used this algorithm to explore various examples of the behaviour of RBNs as certain parameters are varied.

Visualisation of the dynamics helps to prime intuition, and to suggest hypotheses to explore. Some conjectures have been posed; more such conjectures could be generated from larger numbers of examples; some of these may be worthy of further investigation.

Acknowledgements

Thanks to Leo Caves, Adam Faulconbridge, Julian Miller, and Jon Timmis for comments on an earlier draft.

References

1. Drossel, B.: Random boolean networks. In: Schuster, H.G. (ed.) Reviews of Nonlinear Dynamics and Complexity, vol. 1. Wiley, Chichester (2008) arXiv:0706.3351v2 (cond-mat.stat-mech)
2. Gershenson, C.: Introduction to random boolean networks. In: Workshop and Tutorial Proceedings, ALife IX, pp. 160–173 (2004)
3. Kauffman, S.A.: Requirements for evolvability in complex systems. Physica D 42, 135–152 (1990)
4. Kauffman, S.A.: The Origins of Order. Oxford University Press, Oxford (1993)
5. Wuensche, A.: Genomic regulation modeled as a network with basins of attraction. In: Pacific Symposium on Biocomput., pp. 89–102 (1998)
6. Stepney, S.: Visualising random boolean network dynamics. In: GECCO 2009. ACM Press, Montreal (2009)
7. Tufte, E.R.: The Visual Display of Quantitative Information. Graphics Press (1983)

RBN-World
A Sub-symbolic Artificial Chemistry

Adam Faulconbridge[1], Susan Stepney[2], Julian F. Miller[3], and Leo S.D. Caves[4]

[1] YCCSA, University of York, Heslington, YO10 5DD, UK
asf500@york.ac.uk
[2] susan@cs.york.ac.uk
[3] jfm7@ohm.york.ac.uk
[4] lsdc1@york.ac.uk

Abstract. We describe a composable dynamical system that uses the emergent properties of coupled random Boolean networks (RBNs) as a basis for a sub-symbolic artificial chemistry. The approach shows potential for open-ended emergent properties and may lead to a foundation for artificial life.

1 Introduction

Artificial chemistries [1] (AChem, AC) have been used for investigations into the emergence and/or early development of biological phenomena in an abiotic environment with arguable success at generating systems with multiple levels of emergence. We believe this is because previous artificial chemistries have used symbolic approaches. We propose a sub-symbolic [2] approach based on composable random Boolean networks (RBNs) to produce a system within which self-organizing multi-level structures could emerge.

In symbolic representations each "atom" has no internal structure. In a sub-symbolic representation, the "atoms of meaning" are emergent properties of complex dynamics. An example useage of sub-symbolism is neural networks in the field AI; the learned information emerges from the network structure and the weights of the links, rather than being explicitly encoded in a fixed set of symbols. Sub-symbolic representations allow new, unforeseen, "atoms of meaning" to emerge from the developing system.

2 Sub-symbolic Artificial Chemistry

Artificial chemistries are analogous to real-world chemistry in that indivisible building blocks (atoms) bond together to produce larger structures (molecules). However, real-world atoms have internal structure (e.g. electron shells) that is not incorporated in an artificial chemistry based on symbols. We propose using a sub-symbolic representation to account for this feature.

A sub-symbolic representation suitable for an artificial chemistry should exhibit the following features:

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 377–384, 2011.

- Deterministic and computationally tractable
- Emergent characteristics
- Composability to enable sub-symbolic representations of molecular structures can be constructed
- Upward and downward causation so that low-level changes have the potential to disrupt higher-level structures and *vice versa*

Composability of a rich sub-symbolic representation allows molecular structures, such as functional groups or polymers, to potentially be more than the sum of their parts. In this fashion we hope that analogies to biological structures may emerge: a protein is one entity but it is composite of amino acids, each of which is composite of several functional groups, which are themselves composite of multiple atoms.

Sub-symbolic composability allows reactions between novel structures to occur without the need to specify additional reaction rules. This is important for evolution within an artificial chemistry as it potentially enables open-ended development.

Decomposability is also a desirable feature of a sub-symbolic representation. By allowing interactions at multiple levels of structure, lower-level changes have the potential to alter higher-level structures (e.g. the breakage of bonds where catalysts separate from their products).

There are many possible sub-symbolic representations, and many possible artificial chemistries using them. We have made some arbitrary choices for the representation and reaction rule in order demonstrate proof-of-concept. The system described below is only one example of many possible artificial chemistries using this sub-symbolic framework. In addition, the example reactions represent only a tiny sample of possible reactions within this chemistry.

3 RBN-World: Chemistry

RBNs

Random Boolean networks [3,4,5] (RBNs) are our system of choice for a sub-symbolic artificial chemistry due to their rich dynamical structure[1]. In this work, we use a reaction rule based upon matching cyclelengths and composition of RBNs.

An RBN consists of n nodes synchronously updated in discrete timesteps. Each node in the RBN has: a Boolean state, inputs from k nodes, and a Boolean function that maps the state of its input nodes to its state at the next timestep. We use $k = 2$ for all RBNs described here. Function and initial state of each node are chosen at random.

All nodes in the network simultaneously update their state at time t based on the states of their inputs at time $t - 1$. The state of an RBN is the collection of

[1] RBNs were originally devised as simple models of the genetic regulatory network within a cell. Subsequent work using RBNs has continued this theme and focused on reflecting biological networks. However, here we use RBNs as composable dynamical systems with emergent properties.

states of the nodes. All RBNs have cyclic behaviour, returning to a previous state after sufficient timesteps. The number of timesteps on a cycle is the *cyclelength*. The distribution of cyclelengths is highly skewed with median $\sqrt{n}$, with a long tail to a theoretical (but rarely seen) maximum value of 2^n. This means that discovering a RBNs cyclelength is computationally tractable.

RBNs exhibit sensitivity to noise perturbations [3], i.e. a change in one node may change the behaviour of the entire network, or may do nothing. This gives a structured richness to the system that is rarely found in combination with such simplicity. RBNs have a vast number of possibilities, yet they have a number of emergent properties (e.g. cyclelength) with complex many-to-one mappings.

For use in a chemistry, RBNs need to be be combined and fragmented. Therefore some modifications have to be made to traditional RBNs which are described below.

Some further definitions are needed for an artificial chemistry; *atoms*, *bonds*, *reactions* and *molecules*. Due to space limitations, we can not cover all of the system in depth and therefore only outline the important features.

Atoms

We define an additional feature of RBNs; *bonding sites* to make bRBNs (bonding random Boolean networks). Bonding sites (b) are one or more additional nodes that are each taken as an input by one ordinary node chosen at random (a single ordinary node has at most one input from bonding sites). Bonding sites do not have any inputs; their state is determined by whether they are "filled" or "empty". See figure 1 for two example bRBNs. For the preliminary work described here, $b = 2$ for all bRBNs.

The coupling of bRBNs through bonding sites mean that a reaction can change one input to a single node. Due to the sensitivity of the dynamics of RBNs, the change of state of bonding site on formation of a bond can have a wide range of effects (or none).

Bonds

A bond links two bRBNs. There can be multiple bonds between the same pair of bRBNs. Each bond requires one bonding site within the pair of bRBNs to become "filled", and each "filled" bonding site is associated with only one bond. In the chemistry described here, we require that the two bRBNs linked by a bond must have equal cyclelengths both before and after bonding.

Reactions

To form a bond, we require that the two bRBNs have equal cyclelengths both when the bonding sites are "empty" and when the bonding sites are "filled". We do not require the cyclelength when the bonding site is "empty" to be equal to the cyclelength when the bonding site is "filled". Example structures before and after a reaction are shown in figures 1 and 2 respectively (summarized in figure 3).

If a bond is not formed, it is attempted again with any higher-level structures the pair of bRBNs are part of. This iteration of attempting bonding and retrying

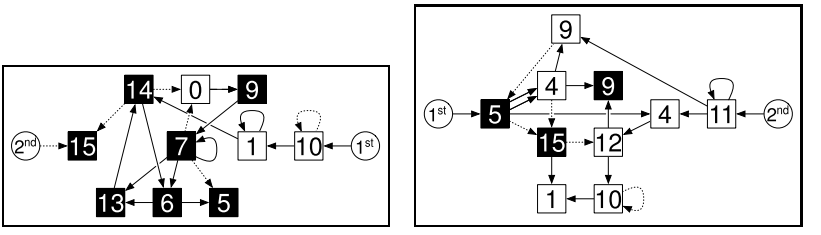


Fig. 1. Two example bRBNs ($n = 10$, $k = 2$, $b = 2$). Numbers are Boolean functions, colour indicates state at this timestep. Edges indicate where outputs are connected to; dashed lines indicate inputs that are always ignored. White circles represent "empty" bonding sites with their bonding order pre-specified.

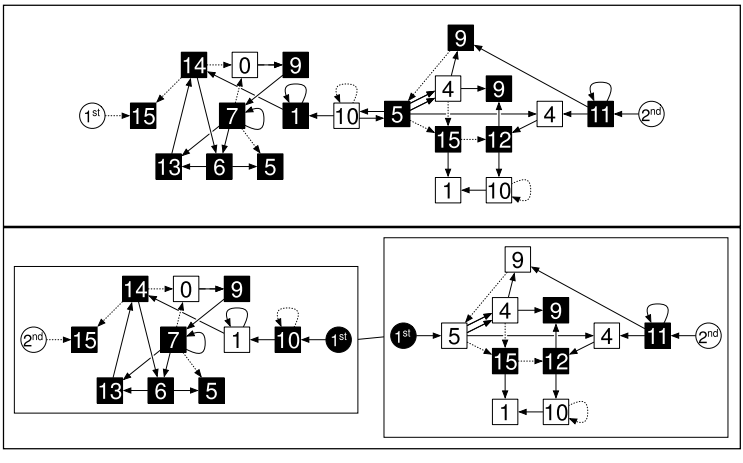


Fig. 2. Example RBN-molecule constructed from RBN-atoms in figure 1. Above is the composite bRBN, and below the component bRBNs. Black circles represent "filled" bonding sites.

for higher-level structures continues until either a bond is formed or there are no more higher structures. See figure 4.

Molecules

bRBNs that are linked by one or more bonds can be expressed as a composite bRBN. The composite bRBN structure is the same as the structure of the component bRBNs, except that inputs from "filled" bonding sites are replaced with direct reciprocated inputs (e.g. figure 2). Non-composite bRBNs are *RBN-atoms*, and a composite bRBN (at any level) is a *RBN-molecule*. RBN-molecules may form bonds in the same manner as RBN-atoms to make higher-level composite structures. In this representation we track multiple levels of structure in order to allow decomposition events. Note that a node can be in different states at different levels of the structural hierarchy.

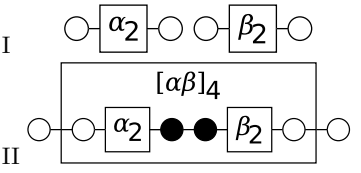


Fig. 3. Abstract representation of figures 1 and 2; RBN-atoms (I) and the RBN-molecule (II). Squares represent the RBN abstracted to a letter and the number shows the current cyclelength. Square brackets denote "is built of" to show that in $[\alpha\beta]_4$ the subscript refers to the combined bRBN rather than just β. The internal structure of a composite bRBN can be similarly expressed, e.g. $[\alpha_2\beta_2]_4$. Note that all RBN-atoms should have square brackets, e.g. $[\alpha]_2$, but for brevity they are omitted for single atoms. The 'lollipops' represent bonding sites; white when "empty" and black when "filled". This reaction can be expressed in symbolic form as $\alpha_2 + \beta_2 \rightarrow [\alpha_2\beta_2]_4$.

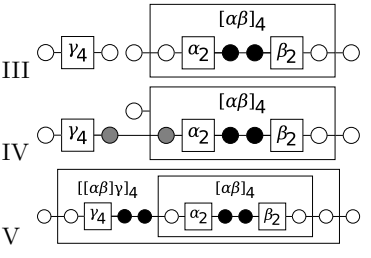


Fig. 4. Example formation of a multi-level structure. The RBN-molecule $[\alpha\beta]_4$ from figure 3 reacts with RBN-atom γ_4. Step III shows the initial condition, and step IV shows the attempted bonding between γ_4 and α_2. The cyclelengths do not match, so they do not bond. The bonding attempt is repeated between γ_4 and $[\alpha\beta]_4$. The cyclelengths do match and step IV shows the forming bond (indicated by the grey 'lollipops'). The final structure is shown in step V. The nested boxes show that γ_4 is bonded to $[\alpha\beta]_4$ rather than α_2. The reaction can be expressed in symbolic form as $[\alpha_2\beta_2]_4 + \gamma_4 \rightarrow [\gamma_4[\alpha_2\beta_2]_4]_4$.

Effects of Bonding

There are two direct consequences to the formation of a bond:

1. The process of bonding changes a bonding site in each linked bRBN from "empty" to "filled". This changes one input to one node, which can potentially lead to a change in cyclelength.
2. The bRBNs linked by the bond form a new higher-level composite bRBN. If one of the participants of the bond was already a component in another bRBN, then the composite structures are combined into a single bRBN.

Additional bonds can be formed as long as the requirements for bonding can be satisfied. Preliminary investigations suggest that complicated structures with multiple levels do form: this requires further investigation to characterise fully.

Bonds can be destroyed as well as created. A bond is broken whenever its two linked bRBNs no longer have equal cyclelengths. The circumstances for this depends on the details of the bRBNs participating in the bond.

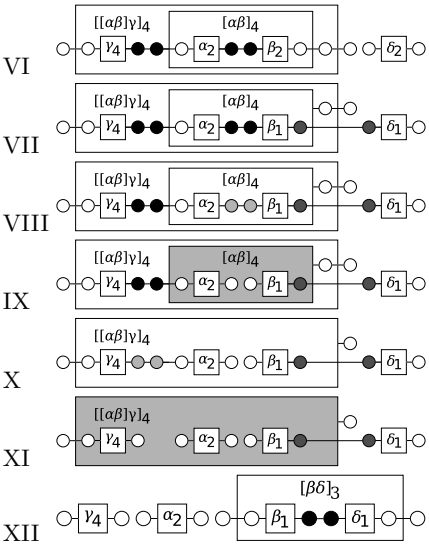


Fig. 5. Continuing from figure 4, δ_2 reacts with β_2. The first stage is to fill in a bonding site on δ_2 and β_2 (shown in dark grey in steps VII – XI above). This changes the cyclelength of β_2 to 1 (see step VII). As β_1 cyclelength is now different to α_2, the bond between them is removed (shown in grey in step VIII). This empties a bonding site in β_1 and α_2. The breakdown of the α_2 — β_2 bond also means $[\alpha\beta]_4$ no longer exists (shown in light grey in step IX) and therefore the γ_4 — $[\alpha\beta]_4$ bond and $[\gamma[\alpha\beta]]_4$ molecule no longer exist (shown in light grey in step X and step XI respectively). The final state at the end of the reaction is shown in step XII above. This reaction can be expressed in symbolic form as $[\gamma[\alpha\beta]]_4 + \delta_2 \rightarrow \gamma_4 + \alpha_2 + [\beta_1\delta_1]_3$.

An example of a reaction that leads to breaking bonds and the decomposition of RBN-molecules can be seen in figure 5. The molecule that has been built up by previous reactions in figures 3 and 4 goes on to react with another RBN-atom. This causes a change in cyclelength which triggers a cascade of bond breakage and structure fragmentation. Processes like these contribute to the rich complex dynamics of this artificial chemistry.

4 Discussion

Composite bRBNs are not identical to conventional RBNs: in addition to the presence of bonding sites, composite bRBNs have a distinctive topology due to the restricted connectivity between the component bRBNs. Here we demonstrate that this does not adversely impact the complex dynamics that we are interested in.

First, we show that the addition of bonding sites to RBNs (and thus the fixing of an input to a node) does not drastically change the distribution of cyclelengths ($\sqrt{n}$ median and long-tail). The results of examining 1,000 RBNs and RBN-atoms over $5 < n < 5000$ are shown in figure 6.

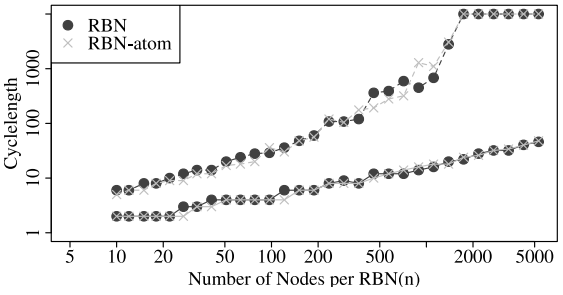


Fig. 6. Distribution of cyclelength in RBNs and RBN-atoms with n nodes. Solid lines are medians, dashed lines the 90$^{\text{th}}$ percentiles; cyclelength was capped at 10,000.

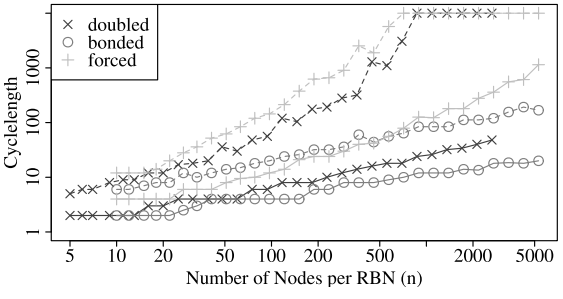


Fig. 7. Distribution of cyclelength for RBN-molecules composite of two RBN-atoms for n nodes. Solid lines are medians, dashed lines the 90$^{\text{th}}$ percentiles; cyclelength was capped at 10,000. "Bonded" refers to atoms that are joined based on matching cyclelengths, "forced" are atoms joined regardless of cyclelength, and "doubled" refers to a single RBN-atom with $2n$ nodes.

Second, we show that RBN-molecules have a similarly shaped distribution of cyclelengths to RBN-atoms. The precise distribution is influenced by the details of the bonding scheme: the requirement for equal cyclelengths, and the topology of composite bRBNs. Therefore, we compare higher-level bRBNs formed in three different ways: bonding between two bRBNs with matching cyclelengths, forced bonding between two bRBNs without any requirements, and a single bRBN with twice the number of nodes, $2n$.

The results of examining 1,000 bRBNs at 30 different values where $5 < n < 5000$ are shown in figure 7. All three bonding schemes result in a broadening distribution with increasing n, though the rate of increase varies. Forced bonding has the steepest increase; this is most likely due to a "lowest common multiple" effect rather than the bond itself. If bond itself has no effect on cyclelength, then the composite structure must have a cyclelength equal to the lowest common multiple of its component bRBNs cyclelengths. If the two bRBNs have the same cyclelength, then the composite structure must also have that cyclelength.

These bonding schemes produce composite structures with long-tailed distributions of cyclelengths. This shows that bRBN-molecules maintain the interesting dynamical properties of RBNs, and thus provide a basis for future higher-order emergence.

5 Future Work

The artificial chemistry described here is a first step in exploring the emergent properties of composable discrete dynamical systems. We note that this framework allows for the specification of whole classes of new artificial chemistries. Some ideas for future work include:

- Varying n, b, and k.
- Limiting Boolean function sets, e.g. no fixed functions
- Characteristics other than cyclelength for bonding
- Requirements other than matching for bonding
- Locating bonding sites by emergent dynamical features (such as node activity) rather than pre-specifying them
- Using other dynamical systems, eg cellular automata (CAs), as atoms
- Identifying a small subset of networks as 'elements'; selected by, for example, a genetic algorithm
- Adding a measure of bond strength, allowing stronger bonds to replace weaker ones
- Introducing spatial aspects
- Introducing thermodynamics and / or entropy as implicit or explicit measures

We have introduced an artificial chemistry based on composable dynamical systems which offers the prospect of rich emergent properties with the potential for open-ended behaviour. A key aspect of our approach is the composition of sub-symbolic components into hierarchical structures, eschewing the need for additional externally imposed rules and / or symbols at each level of organisation. Here an illustrative RBN-based artificial chemistry has been used for proof-of-concept, but other dynamical systems and interaction schemes are possible. We propose that sub-symbolic composable systems provide a framework for the open-ended evolution of artificial life with emergent features.

References

1. Dittrich, P., Ziegler, J., Banzhaf, W.: Artificial chemistries — a review. Artificial Life 7(3), 225–275 (2001)
2. Smolensky, P.: On the proper treatment of connectionism. Behavioral and Brain Sciences 11(1), 1–74 (1988)
3. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22(3), 437–467 (1969)
4. Kauffman, S.A.: The Origins of Order. Oxford Univ. Press, Oxford (1993)
5. Drossel, B.: Random Boolean Networks. Reviews of Nonlinear Dynamics and Complexity, vol. 1, pp. 69–110. Wiley-VCH, Chichester (2008)

Flying over Mount Improbable

Pietro Speroni di Fenizio[1], Naoki Matsumaru[2], and Peter Dittrich[2]

[1] Evolutionary and Complex System Group
Engenheria Informatica,
Coimbra University, Coimbra, Portugal
speroni@dei.uc.pt
[2] Bio Systems Analysis Group and Jena Centre for Bioinformatics
Department of Mathematics and Computer Science
Friedrich-Schiller-University Jena, Jena, Germany
{naoki,dittrich}@minet.uni-jena.de

Abstract. Using Chemical Organisation Theory [1] we present here an analysis of two classical models of artificial chemistries: a system equivalent to AlChemy [2], and the Automata Chemistry [3]. We show that Chemical Organisation Theory is able to explain why AlChemy was unable to evolve, while the Automata Chemistry would produce a stream of novelty that would on the one side explore the space of the possible molecules (and organisations) and on the other build upon the previous findings of the system. We relate to Suzuki's et al. [4] ten necessary conditions for the evolutions of complex forms of life, by adding an 11th one.

1 Introduction

One of the key models that was presented at the beginning of Artificial Life field was AlChemy [2]. AlChemy, which stands for Artificial Chemistry, was used both to suggest a chemical beginning of life [5] (different from the RNA beginning of life, and different from the fat beginning of life), and to explain how our tools to study complex systems were in fact really blunt. It was explained how we were able through an Ordinary Differential Equation (ODE) to study a system which had already all the elements present, but we were not really able to handle a system where new components were being produced [6]. A kind of system were novelty was being generated, in the form of new components, was then called a Constructive Dynamical System.

Twenty years later both threads of research are still alive. Constructive Dynamical System theory gave rise to Chemical Organisation Theory, which expands it, using Algebra, to deal with more general systems. It still studies artificial chemistries, but more generally deals with Reaction Networks, and had been used successfully in bioinformatics, and systems biology to predict and describe the algebraic structure of various chemical systems, from the atmosphere in Mars and Io [7], to the internal metabolism of a unicellular being [8].

Artificial Chemistry, as a research tool, has been used in the study of protolife. The AlChemy system was observed not to spontaneously evolve, and thus

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 385–393, 2011.

researchers turned their attention to other artificial chemistries. But no one ever answered, or even tried to answer, why was AlChemy unable to evolve, and what additional lessons can we gain from this.

We shall use in this regard Chemical Organisation Theory, so the same theory that was presented using AlChemy, will now, in its more mature form, be applied back to study a system equivalent to AlChemy, finally explaining why was AlChemy unable to produce an evolving system. We will compare this with the study of another Artificial Chemistry, the Automata Chemistry model, and showing how there, instead, we do observe a genuine evolution. So the system keep on being constructive, keeps on producing novelty, and exploring the space of possible organisations. In 2003 a paper was written that listed ten on the necessary conditions for the Evolution of Complex Forms of Life in an Articial Environment. Those conditions were [4]:

1. the symbols or symbol ingredients be conserved (or quasiconserved) in each elementary reaction, or at least, conserved with the aid of a higher-level manager.
2. an unlimited amount of information be coded in a symbol or a sequence of symbols.
3. particular symbols that specify and activate reactions be present.
4. the translation relation from genotypes to phenotypes be specied as a phenotypic function.
5. the information space be able to be partitioned by semipermeable membranes, creating cellular compartments in the space.
6. the number of symbols in a cell can be freely changed by symbol transportation, or at least can be changed by a modication in the breeding operation.
7. cellular compartments mingle with each other by some randomization process.
8. in-cell or between-cell signals be transmitted in some way like symbol transportation.
9. there be a possibility of symbols being changed or rearranged by some randomization process.
10. symbols be selectively transferred to specic target positions by particular activator symbols (strongly selective), or at least selectively transferred by symbol interaction rules (weakly selective).

To those ten conditions we will add an eleventh:

11. The system should not have access to a basis that permits the construction of every possible molecule.

We will then discuss the consequences of this new condition, and it's relations with the previous ten. To do all this we shall first briefly present Chemical Organisation Theory, the Combinator's Alchemy version, and the Automata Chemistry. We will then show the result that we obtained by applying the Chemical Organisation Theory to the Combinators Chemistry, and to the Automata Chemistry. We will then discuss those results, and reach some conclusions.

2 Description of Chemical Organisation Theory

Chemical Organisation Theory has been presented in multiple papers, especially in previous versions of this conference. Although the results that we are presenting here are new, the actual theory has not changed. The theory in its complete format can be found from [1]. We shall now only repeat a brief description. Please note that the Artificial Chemistries we will study here are not the most general artificial chemistries possible, but part of a very specific type of systems called Catalytic Flow Systems. Those systems are often studied in Artificial Life, where as more general system are usually present in biology. We consider an artificial chemistry as a set of molecules M and a function R called reaction. R will be a function of arity 2 from $M \times M$ to M (i.e. $\forall\, x, y \in M$, $R(x, y) \in M$). We define an organisation as a set of molecules which is both closed and self maintaining. That is, let O be such a set, for all $a, b \in O$, $R(a, b) \in O$ (closure). And for all $c \in O$, such that c can be destroyed, there exist $a, b \in O$ such that $R(a, b) = c$ (self maintenance).

Note that this description is similar to the one given by Fontana in 1992 [2]. The only difference, at this stage, is that Fontana's self maintenance was required for every molecule, and not just for each molecule that can be destroyed (either through an out-flux or through a non catalytic reaction). This seemingly small difference is necessary to permit to the theory to study systems where some molecules interact in a catalytic way with every other molecule, and are not subject to an outflux (or destruction process). This is necessary, for example, to model DNA molecules in a biological system. In our simplified systems this difference makes sure that if the system reaches a configuration where no reaction is possible, then the configuration is also (trivially) an organisation.

The organisations generated by an artificial chemistry, form a partially ordered set (ordered by the inclusion), and more precisely form a lattice LO. Also it is possible to define a function GO(S) that given a set returns the organisation generated by that set. So organisations partition the space of all possible sets, and as the system travel in the space of possible molecules, we can follow it on LO. All this becomes important as the system evolves; in fact the evolution of the system will be, mathematically, represented as a movements on the lattice of organisations. All those results were previously presented in [1] [9] [10]. Briefly we could say that in this paper we are studying and comparing the movement in the lattice of organisations of two different systems.

If a system is left to react to itself, if any change is present, this will always be toward simpler organisations, that is toward organisations laying lower in the lattice of organisations (downward movement). But when the system is seeded with random molecules, those molecule can push the system toward a more complex organisations (upward movement). And if this is unstable fall back in the same organisation, or on a neighbouring organisation (sideward movement). We can now see evolution as an interaction between the random variation which leads the system toward a state of greater complexity, and its simplification by reaching a stable subsystem.

3 Systems: The Combinators AlChemy Version

The first system that we have applied the Chemical Organisation Theory to is the Artificial Chemistry generated by combinators. For a complete description please refer to [11,12]. For those experiments we will use a simplified version of the system which has no R molecule, only catalytic reactions, and no fixed amount of basic atoms. So a system which is, in all regards, equivalent to Fontana's AlChemy except that it used Combinators, instead of Lambda terms. For a study on how combinators are equivalent to Lambda terms please refer to [13]. Please note, for example, that we could observe the same organisations that Fontana observed. And analyse them from a Chemical Organisation point of view ([10], chapter 6).

We will briefly describe the system. The system is an Artificial Chemistry, whose molecules are combinators in their normal form. Briefly we can say that Combinators are a string with balanced parenthesis over an alphabet of basic operators. Strict rules define how the basic operators in the string are applied, thus a combinator end up being the operator that is produced by the joined reaction of all the operators that compose it. The result is thus an operator, which can be applied to a string with balanced parenthesis, and would then produce a new string (again with balanced parenthesis). So a combinator is an operator which applied to a combinator generates another combinator.

Some combinators are in an unstable configuration, and by applying the operators that compose them, can change their configuration. If this can happen we say that a combinator is not in its normal form. The process that transform a combinator into another is called reduction. A fundamental theorem, in combinator theory, is that if a combinator can be reduced in a normal form, this is unique. In our experiment we shall use only combinators in their normal form, and when the result of a reaction is a new combinator, we shall just permit consider the reaction to be valid if the result can be reduced (i.e. if we could find in t steps a reduction) to a combinator in its normal form.

A family of combinators (called Soup) are present in the experiment, and at each time-step two combinators are randomly chosen, interacted, and if the result is a combinator that can be reduced to a normal form, the result is added to the soup. A random combinator is then eliminated. The basic alphabet that were used were (B, C, K, I, S, W) which include two basis of the space of combinators (B, C, W, K) and (K, S, I). Their behaviour as operators can be found both in [12] and in [13].

4 Systems: The Automata Chemistry

The second artificial chemistry used was an automata chemistry [3]. In this chemistry molecular species are binary strings ($s \in \{0,1\}^{32}$) with a constant length of 32 bits. As in the other chemistry, two strings will catalyze the production of a third string ($s_1 + s_2 \rightarrow s_3$). One of the strings s_1 is mapped to an automaton A_{s_1} according to a well dened instruction table (we used code

table II in [3] allowing self- well defined instruction table (we used code table II from [3] allowing self- replication). The other, s_2, serves as input to A_{s_1}, and the result of the reaction is the output of the automaton $s_3 = A_{s_1}(s_2)$. In each time step, two string are randomly chosen to catalitycaly react. After the reaction the reactants are inserted back into the reactor while one randomly chosen molecule in the reactor is replaced by the product in order to keep the total number of the objects in the reactor constant at value N.

5 Results

We applied the Chemical Organisation Theory analysis to both the Combinator's AlChemy system, and to the Automata Chemistry. The results that we reached were vastly different.

We could not map the whole lattice of organisations in either systems. In the one case (the combinators) this was infinite. In the other case (the matrix chemistry) while not infinite, it was too vast to be calculated. What instead we did was to stop the system in various points, and study the organisation that was being generated by the molecules present in the Soup. Note that we made multiple runs, and each run of the system was unique. Studying such systems presented a challenge, since the differences from one run to the other were mostly qualitative, before being quantitative. This was true both in terms of the differences between run on the same system (example, two runs of the matrix chemistry), and even more between a run on one system and a run on the other. The organisations, that were generated, the historical trajectory through those organisations were often very different one from the other. In both case it would not make sense to make a statistical analysis of a system. Yet there were some general pattern that could be recognised. Some common ways in which the Automata Chemistry system would run, versus how the Combinators' AlChemy System would run. As such, after having observed a number of runs, we are presenting here data from two exemplary runs. They are in all regards typical run, one of the Combinators' AlChemy System, and one of the Automata Chemistry. We then discuss the differences between the two type of systems.

In the Combinators' AlChemy System, not only we could not draw the lattice of all the possible organisations, but we could often also not map the organisations that were being generated. We note that since the system possessed a simple basis (two in fact: S, K; B, C, W, K), it was possible to have a configuration of molecules in the Soup that could potentially generate the whole system. Every possible molecule could be generated (given enough time, and a Soup big enough) by the system in such configuration. As such the organisation generated by the system, in those configuration, was potentially the whole system. And every possible other organisation that existed was a subset of this. We shall call this the organisation Infinity. As the system would keep on reacting, eventually the molecules of the basis would be destroyed, and the system would move downward to a smaller lattice, until eventually it would produce a finite lattice. When the system, was finally simple enough, we could study it with Chemical organisation

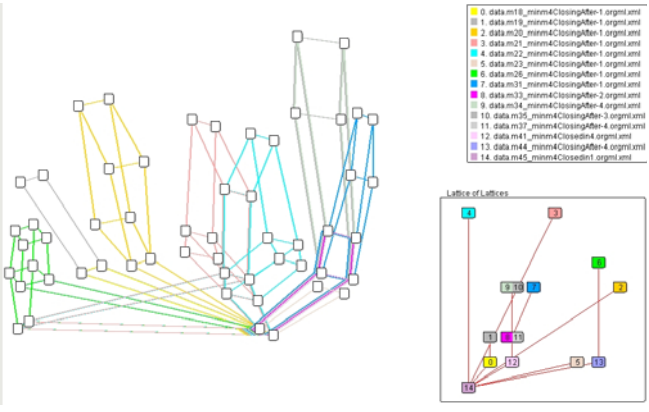


Fig. 1. This is a figure showing the evolution of the molecules in time, above. And the various organisations that suceeded below.

Theory, and we could map its lattice. Although we would limit ourselves to the point in time where the system was simpler, often the system would still be too complex to be nailed down through Chemical Organisation Theory. Usually to study the lattice of the organisations we start by calculating the largest possible organisations. This is done by starting with a set of molecules M, then producing every molecule that can be generated by reacting every pair of molecules together, thus generating M^1. Then repeating the same process taking pairs in M^1 we generate M^2, and so on. Until $M^n = M^{n-}1$. And then the system is contracted to the biggest self maintaining set. (For a complete description of the process please refer to [10], Chapter 2. In all but the most simple cases there was no n such that $M^n = M^{n-1}$. As the $lim_{n->\infty}|M^n| = \infty$. And we often had to limit ourselves to n such that $|M^n| < t$ (with the threashold often = 600).

We know we were very abruptly simplifying the lattice of organisations generated,losing potentially significant data. Still the data that we could collect were interesting, and pointed to some fundamental differences between the two systems.

As the system would keep on reacting, eventually the molecules of the basis would be destroyed, and the system would start to generate a smaller lattice, until eventually it would produce a finite lattice. When the system, was finally simpler enough, we could study it with Chemical organisation Theory, and we could map its lattice. What we would observe is that the system would jump from one organisation to the other. With no sense of historical continuity. Although we recognise that the historical continuity might be present in the data that we could not analyse, much of those data contained the full lattice. And as such the system was essentially going from organisation A to Organisation Infinity, to organisation B, to Organisation Infinity, to organisation C, to organisation Infinity, etc...

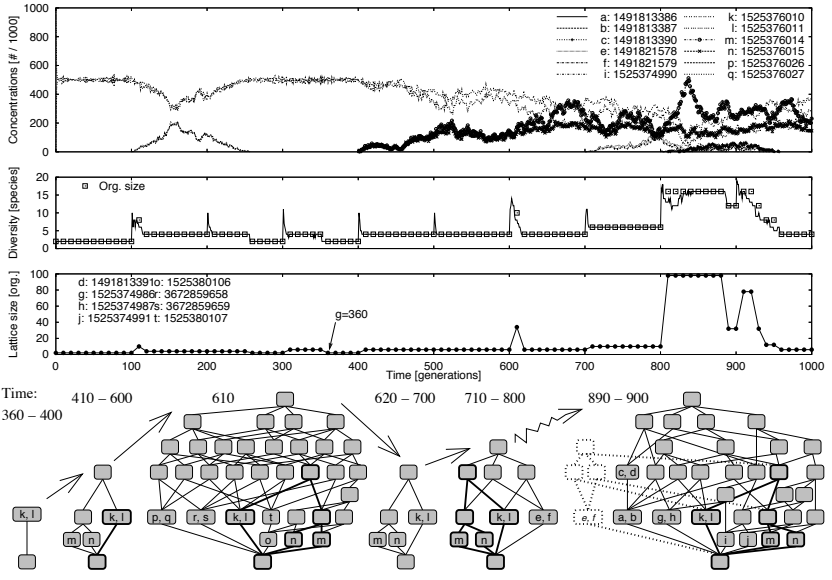


Fig. 2. This is a figure showing the evolution of the molecules in time, above. And the various organisations that suceeded below. (adopted from [9]).

When we applied Chemical Organisation Theory to the Automata Chemistry the results were totally different. In this case it was possible to calculate the lattice of all the possible molecules that could be generated by the system.

In this case, not only we could observe the system's organisation in every instant, but we could observe how the random molecule would push the system into a more general organisation, and from there how the system would reach a simpler, but more stable organisation. The net result was a system that would reach an organisation, expand into a more complex, but unstable one. From there either collapse back toward the same organisation, or reach a different organisation. Thus producing a novel behaviour which was generally either an expansion of the previous one, or a partial modification of it.

6 Discussion

What is really striking between those two systems is how different is their evolving process. In a sense both systems are very similar. They both are produced by many molecules (232 in one case, infinite, but actually limited by the memory of the computer in the other), the reaction is equivalent, they both use catalytic reactions, with an out-flux of a molecule every time a new molecule is produced. The size of the experiments were similar (both used Soups of a thousand molecules). Both systems had random molecules being inserted, and in both cases the speed of the insertion was chosen so the system had the time

to settle in an organisation before new random molecules were inserted. And yet the evolutive behaviour was totally different.

In the first case the system would reach a finite organisation, then would wait until a random molecule would push it away. Then it would move into a organisation which was too vast to be studied. Often the soup would contain a basis of the set of all molecules, and thus the generated organisation would be the organisation Infinity. From there the system would move in an unpredictable way, eventually losing the key molecules that could potentially generate the wider organisations. And from there it would move down, to a new organisation. The new organisation would most often have no relation to the previous one. As such the system was similar to a system that was randomly picking organisations from the lattice of all possible organisations. With little or no relation to the previous organisation present in the system. Although this system is effectively moving from one organisation to the other, it was unable to hold build upon previous subsystems discovered.

The second case was very different. First of all we were always able to calculate the organisation generated by the molecules in the soup. Then the set would grow slowly. Often even under the influence of random noise the system would remain unchanged. Then when it would change it would move toward a more complex system (after having incorporated the new molecules), and then drop from there to a simpler system, which sometimes was the original one, and sometimes it was not. There was a very definite continuity from one state of the system to the other. And we could see the system exploring the lattice of organisations, moving through neighbouring organisations.

7 Conclusions

Often in Artificial Life there is a constant search for the most powerful system. The system that can potentially produce a bigger, higher complexity. It is inside this line of thought that AlChemy was developed. AlChemy having universal computation capabilities (U.C.C.) was able to produce every possible lambda term. Thus every possible subsystem would fall in its domain. Yet in this case it is this very power that gets in the way toward a genuine evolutive search of the space of possibilities. For each combinator that is present a counter combinator is possible that can destroy it. And the result is that no organisation is able to be stable enough. The problem is not just with the potential capabilities of the system (the fact that it has U.C.C.), but that a basis was also present. By taking lambda terms Fontana (and then combinators, one of us) was using a system that has been explored by mathematicians for close to a century. In mathematics there is a constant search for the most elegant (i.e. shortest) basis of a system. Thus the basis B, C, W and S, K were developed. By inserting in the system random molecules, composed of those basic atomic structures, the system produced was effectively able to reach too easily the infinite organisation. Per contro, we do not know what is the basis of the Automata Chemistry. Although we know that it exists, we also know that the random molecules that we were inserting in that

system were not often containing elements of the basis (or we would be seeing a much wider organisation appearing). So the system had to explore the space, with no shortcut that could let it easily get rid of molecules that were present. We recall in this regard another historical model, Tierra. Tierra had universal computations capabilities, but the basis was not so elegantly expressed. And Tierra evolutive behaviour was more similar to the Automata Chemistry, with its slow progress, than to AlChemy. We thus conclude that an important element in the construction of a system able to evolve is the absence of a basis of the whole system among the basic building blocks with which the system is fed when random molecules are inserted. Although the basis must necessarily be present, it should not appear too easily.

References

1. Dittrich, P., Speroni di Fenizio, P.: Chemical organisation theory. Bull. Math. Biol. 69(4), 1199–1231 (2007)
2. Fontana, W.: Algorithmic chemistry. In: Langton, C.G., Taylor, C., Farmer, J.D., Rasmussen, S. (eds.) Artificial Life II, pp. 159–210. Addison-Wesley, Redwood City (1992)
3. Dittrich, P., Banzhaf, W.: Self-evolution in a constructive binary string system. Artif. Life 4(2), 203–220 (1998)
4. Suzuki, H., Ono, N., Yuta, K.: Several necessary conditions for the evolution of complex forms of life in an artificial environment. Artif. Life 9(2), 153–174 (2003)
5. Fontana, W., Buss, L.W.: 'The arrival of the fittest': Toward a theory of biological organization. Bull. Math. Biol. 56, 1–64 (1994)
6. Fontana, W., Buss, L.W.: The barrier of objects: From dynamical systems to bounded organization. In: Casti, J., Karlqvist, A. (eds.) Boundaries and Barriers, pp. 56–116. Addison-Wesley, Redwood City (1996)
7. Centler, F., Dittrich, P.: Chemical organizations in atmospheric photochemistries - a new method to analyze chemical reaction networks. Planetary and Space Science 55, 413–428 (2007)
8. Centler, F., Speroni di Fenizio, P., Matsumaru, N., Dittrich, P.: Chemical organizations in the central sugar metabolism of escherichia coli. In: Mathematical Modeling of Biological Systems, vol. I. A Birkhäuser book, Basel (2007)
9. Matsumaru, N., Speroni di Fenizio, P., Centler, F., Dittrich, P.: On the evolution of chemical organizations. In: Artmann, S., Dittrich, P. (eds.) Explorations in the Complexity of Possible Life: Abstracting and Synthesizing the Principles of Living Systems, Proceedings of the 7th German Workshop of Artificial Life, pp. 135–146. Akademische Verlagsgesellschaft Aka GmbH, Berlin (2006)
10. Speroni di Fenizio, P.: Chemical organization theory. PhD thesis, Friedrich-Schiller-University Jena (February 2007)
11. Speroni di Fenizio, P.: A less abstract artficial chemistry. In: Bedau, M.A., McCaskill, J.S., Packard, N.H., Rasmussen, S. (eds.) Artificial Life VII, pp. 49–53. MIT Press, Cambridge (2000)
12. Speroni di Fenizio, P., Banzhaf, W.: Stability of metabolic and balanced organisations. In: Kelemen, J., Sosík, P. (eds.) ECAL 2001. LNCS (LNAI), vol. 2159, pp. 196–205. Springer, Heidelberg (2001)
13. Hindley, J.R., Seldin, J.P.: Lambda-Calculus and Combinators: An Introduction. Cambridge University Press, New York (2008)

Behaviors of Chemical Reactions with Small Number of Molecules

Yasuhiro Suzuki

Nagoya University, Furocho Chikusaku Nagoya city Aichi 464 8603 Japan
ysuzuki@is.nagoya-u.ac.jp
http://sci.cs.is.nagoya-u.ac.jp

Abstract. Living systems are composed of biochemical reactions and many of them involves a small number of molecules. We investigate the behaviors of chemical reactions of the Lotka-Volterra model with small number of molecules by using Abstract Rewriting System on Multisets, ARMS; ARMS is a stochastic method of simulating chemical reactions and it is based on the reaction rate equation. We confirmed that the magnitude of fluctuations on periodicity of oscillations becomes large, as the number of involved molecules is getting smaller and the dynamical characteristics is changed. We investigate the coarse grained state space of ARMS and show that the mechanism of fluctuations occur in the chemical reactions involved a small number of molecules.

Keywords: Artificial Chemistries, Lotka-Volterra Model, Small Number Effects, Chemical reactions with a small number of molecules.

Introduction

In biochemical reactions in living systems involve a small number of molecules; for example Transcription from DNA to RNA in the cell involves a small number of messenger RNA (mRNA) molecules and two copies of each gene. For such chemical reactions with a small number of molecules, stochastic effects must be considered.

Gillespie[2] proposed a stochastic method of simulating chemical kinetics, which has firmer physical basis than the deterministic formulation such as the differential equations. Such stochastic chemical kinetics as the chemical master equation, is often mathematically intractable. So the Monte Carlo procedure is often used to simulate; since it requires a great amount of computer time, several approximate procedures have been proposed [3], [13].

Abstract Rewriting system on MultiSets, ARMS was proposed[7] as an *Artificial Chemistry*[1], [4]. ARMS have been used in the Artificial Life, for example, for considering the mechanism of *edge of chaos*[9], modeling chemical evolution in the origin of life[9] and an evolutional reaction network [10]. Beyond the Artificial Life, ARMS have been used in various subjects: the Systems biology (modeling the P53 signaling network [11], inflammatory response [12]), the Physical chemistry (modeling the *Belouzov-Zhabotinskii* reaction [9], [13]), Chemical Ecology [9] and so on.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 394–401, 2011.

Abstract Rewriting System on Multisets, ARMS

ARMS is a construct $\Gamma = (A, w, R)$, where A is an alphabet, w is the initial state and R is the set of reaction rules.

Let A be an *alphabet* (a finite set of abstract symbols). A *multiset* over A is a mapping $M : A \mapsto \mathbf{N}$, where $\mathbf{N}$ is the set of natural numbers; 0, 1, 2,.... For each $a_i \in A$, $M(a_i)$ is the *multiplicity* of a_i in M, we also denote $M(a_i)$ as $[a_i]$. We denote by $A^{\#}$ the set of all multisets over A, with the empty multiset, $\emptyset$, defined by $\emptyset(a) = 0$ for all $a \in A$. A multiset $M : A \mapsto \mathbf{N}$, for $A = \{a_1, \ldots, a_n\}$ is represented by the state vector $w = (M(a_1), M(a_2), \ldots, M(a_n))$, w. The union of two multisets $M_1, M_2 : A \mapsto \mathbf{N}$ is the addition of vectors w_1 and w_2 that represent the multisets M_1, M_2, respectively. If $M_1(a) \leq M_2(a)$ for all $a \in A$, then we say that multiset M_1 is included in multiset M_2 and we write $M_1 \subseteq M_2$.

A *reaction rule* r over A is defined as a couple of multisets, (s, u), with $s, u \in A^{\#}$. A set of reaction rules is expressed as R. A rule $r = (s, u)$ is also represented as $r = s \rightarrow u$. Given a multiset $w \subseteq s$, the application of a rule $r = s \rightarrow u$ to the multiset w produces a multiset w' such that $w' = w - s + u$. Note that s and u can also be zero vector (empty).

The *reaction vector*, ν_j denotes the change of the number of molecules produced by the rule r_j. For example ν for the reaction $a, b \rightarrow b, c$ is $(-1, 0, 1) \equiv (a, b, c)$. We employ multisets; such a multiset $X : A \mapsto \mathbf{R}$ for $A = \{a_1, \ldots, a_n\}$ is represented by the state vector $\mathbf{x} = (X(a_1), X(a_2), \ldots, X(a_n))$. $X(a_i)$ denotes the number of specie a_i. Let us assume that there are $N \geq 1$ molecular species $\{a_1, ..., a_n\}, a_i \in A$ that interact through reaction rules $R = \{r_1, ..., r_m\}$. As the time evolution of $\mathbf{x}$ unfolds from a certain initial state, let us suppose the state transition of the system to be recorded by marking on a time axis the successive instants $t_1, t_2, ...$ as $X(t_j)$ $(j = 1, 2, ...)$, where $j = 0$ denotes the initial state. We specify the dynamical state of $\mathbf{x}(t) \equiv (X(a_1(t), X(a_2(t)), ..., X(a_N(t)))$, where $X(a_i(t))$ is the number of a_i specie at time t. The time evolution of ARMS is given as $\mathbf{x}(t) = \mathbf{x}(t-1) + \nu_j$, when there are several rules can be applied for $\mathbf{x}(t)$, only one rule is selected for applying. We can define various ways of applying rules and in this study rules are applied sequentially according to the mass action law of chemical reactions. We will define it in the next section.

ARMS with Chemical Kinetics

We modify the ARMS for modeling chemical kinetics and assume that all chemical reactions take place in a well-stirred reactor; this assumption is required due to the strong dependence of the reaction rate on the concentration of the reagent species. For the dynamical state $\mathbf{x}$(t), we define the probability of selecting $\nu_j \equiv P_{\nu_j}(\mathbf{x}(t))$ as

$$P_{\nu_j}(\mathbf{x}(t)) = \frac{c_j(\mathbf{x}(t)) \times k_j}{\sum_{j=1}^{m} c_j(\mathbf{x}(t))}, \tag{1}$$

where $c_j(\mathbf{x}(t))$ denotes the number of possible combination collisions of r_j reactant molecules on $\mathbf{x}(t)$, k_j is the reaction constant of r_j. The time evolution of

$\mathbf{x}(t)$ is a jump Markov process on the N-dimensional non-negative lattice. We define the function $f(\mathbf{x}(t))$, called the *propensity function* for $r_j \in R$ on $\mathbf{x}(t)$ by $f(\mathbf{x}(t)) = (P_{\nu_1}(\mathbf{x}(t)), P_{\nu_2}(\mathbf{x}(t)), ..., P_{\nu_m}(\mathbf{x}(t)))$.

Lotka-Volterra model. The Lotka-Volterra model, LV describes interactions between two species in an ecosystem, a predator and a prey. Reaction rules of the LV are given as;

$$\begin{aligned} X &\xrightarrow{a} X, X : (r_1), \\ X, Y &\xrightarrow{b} Y, Y : (r_2), \\ Y &\xrightarrow{c} \perp : (r_3), \end{aligned}$$

where a b and c are reaction constants and $\perp$ denotes an empty symbol, which represents decay of Y.

The reaction rate equation, RRE of the LV is

$$\dot{X} = aX - bXY = X(a - bY) \tag{2}$$

$$\dot{Y} = bXY - cY = Y(bX - c). \tag{3}$$

Since population equilibrium occurs in the model when neither of the population levels is changing, from $X(a - bY) = 0$ and $Y(bX - c) = 0$, equilibria of the LV are $X = Y = 0$ and $Y = a/b$, $X = c/b$. The first solution represents the extinction of both species and the second solution represents a equilibrium point at which both populations sustain their current, non-zero numbers. The LV shows oscillations around this equilibrium point.

We analyze the behavior of the LV around the equilibrium point. The RRE of the LV can be rewritten into

$$\left(\frac{c}{x} - d\right)\dot{x} + \left(\frac{a}{y} - b\right)\dot{y} = 0,$$

and we obtain

$$\frac{d}{dx}\left[c\log x - dx + a\log y - by\right] = 0.$$

We define

$$H(x) = \bar{x} - x, \quad G(y) = \bar{y}\log y - y,$$

where $\bar{x} = c/d$ and $\bar{y} = a/b$. Then we obtain the Lyapnov function,

$$V(x, y) = dH(x) + bG(y).$$

Since

$$\frac{d}{dt}V(x(t), y(t)) = 0, \quad V(x(t), y(t)) = constant,$$

the time evolution of LV is periodic. Hence if fluctuations displace the time evolutions from the equilibrium point, the system should show periodic time evolutions and these fluctuations never lead the time evolutions to the equilibrium point [6]. Hence, we will observe a variety of periodic oscillations, when the magnitude of fluctuations are large.

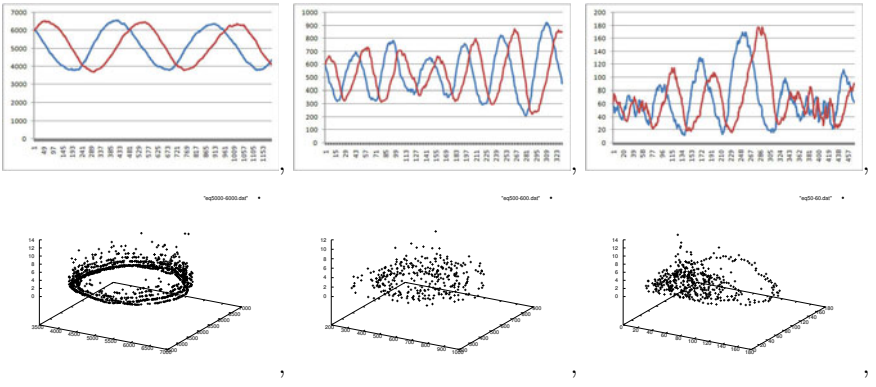


Fig. 1. The time evolution and its phase space and the distribution of existence probability are arranged one above the other; from left to right, the equilibrium points and initial states are (5000, 5000) and (6000,6000), (500,500) and (600, 600) and (50,50) and (60, 60); the Z-axis denotes the value of the existence probability respectively

Comparison of Schematic Views

We compared the periodic oscillations by changing equilibria; we examined the case when equilibria $(X, Y) = (5000, 5000)$, $(500, 500)$ and $(50, 50)$, for each equilibrium point we set the initial state as $(X, Y) = (6000, 6000)$, $(600, 600)$ and $(60, 60)$, respectively (top row in the fig.1).

We confirmed that the magnitude of fluctuations on periodicity of oscillations became large, as the equilibrium point was getting smaller. When the equilibrium point $(X, Y) = (5000, 5000)$, the magnitude of fluctuations was small and when $(X, Y) = (500, 500)$, fluctuations became large, while keeping periodic oscillations. When $(X, Y) = (50, 50)$, fluctuations became more larger and a variety of oscillations were observed; especially small period oscillations near the equilibrium point was observed.

Phase Space

We compared phase spaces near to the each of equilibrium point (bottom raw in the fig.1) and existence probability; the existence probability of $\mathbf{x}_i \in \mathbf{x}(t), t = 1, 2, ...$ was obtained by dividing the number of times of visiting $\mathbf{x}_i$ by the total number of visiting states in the time evolution. When the equilibrium point $(X, Y) = (5000, 5000)$, cyclic phase structure was clearly observed and the distribution of existence probability was mostly homogenous around the cyclic phase structure. When $(X, Y) = (500, 500)$, a variety of cyclic phase structures were observed; the distribution of existence probability was mostly homogeneous and these structures generate a variety of periodic oscillations. When $(X, Y) = (50, 50)$, the phase structure was skewed; the distribution of existence probability was inhomogeneous and most of them were attracted near to the equilibrium point.

Coarse Graining of Probabilistic Field

From comparison of schematic views, it was shown that the number of molecules in reactions effect behaviors of the time evolutions. Next, we will focus on effect of the number of molecules in reactions to the probability of selecting reaction rules. Since the probability of selecting a reaction rule was given by the propensity function on $\mathbf{x}$, we examine the probabilistic field of the propensity function. The propensity function of the LV was $f(\mathbf{x}) = (\frac{aX}{M}, \frac{bXY}{M}, \frac{cY}{M})$, where $M = aX + bXY + cY$.

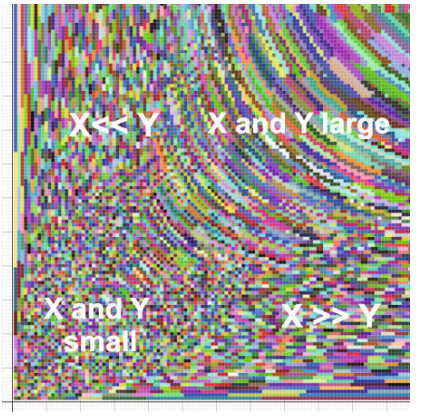


Fig. 2. The coarse grained probabilistic field of the LV; the equilibrium point was $(X, Y) = (10, 10)$ and from $(1, 1) to (200, 200)$ were observed, where the same color illustrates the same closure

Closure: A closure is defined as a sub space in a state space, where every value of a propensity function is approximately the same. The closure of $\mathbf{x}_i$ is defined as the set of vectors $\mathbf{x}_j, (j = 1, 2, 3, ...)$, where the maximal absolute values in the differences of vector $\mathbf{x}_i$ and $\mathbf{x}_j$ are less than ϵ, ϵ is a very small number and given in advance; $\{ \mathbf{x}_j \mid MAX \mid f(\mathbf{x_j}) - f(\mathbf{x_i}) \mid < \epsilon\}$, where the function MAX($\mathbf{x}$) returns the maximal value in $\mathbf{x}$. We investigated the coarse grained phase space of equilibrium point $(X, Y) = (10, 10)$, while $(X, Y) = (1, 1)$ to $(200, 200)$. When the value of X and Y were small, the size of closures was 1 and the size of a closure was proportion to the value of X and Y. The shape of closures were different according to the value of X and Y; when X and Y were large, the shape of closures were hyperbolic; when X was considerably larger than Y, the shape of closures were horizontally, on the other hand, when Y was considerably larger than X, the shape of closures were vertically (fig 2). Because, when X or/and Y is/are so large that $\frac{bXY}{M}$ is considerably larger than $\frac{aX}{M}$ and $\frac{cY}{M}$, we can ignore $\frac{aX}{M}$ and $\frac{cY}{M}$ in a propensity function, so the propensity function can be regarded as $f(\mathbf{x}) \simeq (0, \frac{bXY}{M}, 0)$, approximately. Hence, if X and Y are large

then the elements in the closure would fulfill $XY \simeq XY \pm \delta$, where δ is $\frac{\epsilon M}{b}$, so the shape of closure becomes hyperbolic. In case X is large and Y is considerably smaller than X, even if X is changed largely, XY would not change so much. So the shape of closures would along the X-axis and horizontally long. On the other hand, in case Y is large and X is considerably smaller than Y, even if Y is changed largely, XY would not change so much. So the shape of closures would along the Y-axis and vertically long.

Size of closure and time evolutions: If the probability of selecting a rule is homogeneous, the directions of time evolutions from the closure are also same. So when the size of closures are large, even if a time evolution fluctuates and bifurcates into several states, those of bifurcated states are likely to be covered by a closure and the directions of time evolutions from these bifurcated states would be kept the same. On the other hand, when the size of closure are small, if a time evolution fluctuates and bifurcates into several states, those of bifurcated states are likely not to be covered by a closure and the directions of time evolutions from the states would be different. Therefore, the time evolution of the reactions with small molecules should be suffered large fluctuations. In this study, because of a reaction rule is applied sequentially, a bifurcation may occur, after the time evolution returns to the visited state.

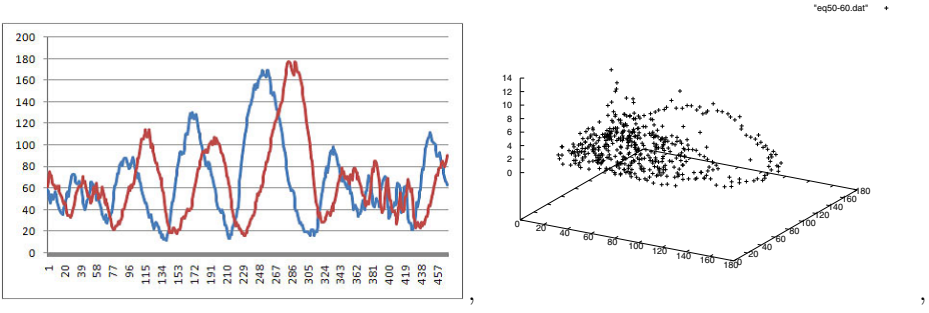


Fig. 3. Small Number Effects: the time evolution was trapped near to the equilibrium point, $(50, 50)$, where the initial state was $(60, 60)$ (during $t = 1 - 77$ and $324 - 437$)

Small Numbers Effects, SNE. We showed that in the LV, fluctuations never leads the time evolution to the equilibrium point; however in reactions with small numbers, we observed the fluctuations lead the system to the equilibrium point (fig. 3). When equilibrium point was $(50, 50)$ and initial state, $(60, 60)$, its time evolution was trapped near to the equilibrium point and oscillated in small periods (in the right of the fig.3). The coarse grained phase space shows that the reactions with $X, Y \simeq 50$, closures are small. So its time evolution would be suffered large fluctuations and trapped. We call the characteristic changes in reactions caused by the small number molecules, the Small Number Effects (SNE). We also observed the SNE in the Brusselator model[5]. The Brusselator is a model of the Belousov Zhabotinskii (BZ) reaction;

$$A \xrightarrow{k_1} X : r_1,$$
$$B + X \xrightarrow{k_2} Y + D : r_2,$$
$$2X + Y \xrightarrow{k_3} 3X : r_3,$$
$$X \xrightarrow{k_4} E : r_4.$$

The Brusselator shows a limit-cycle oscillation; when a time evolution in the limit-cycle oscillation, even if a time evolution fluctuates, it returns to the limit-cycle oscillation and does not show a variety of periodic oscillations, which as we have seen in the LV.

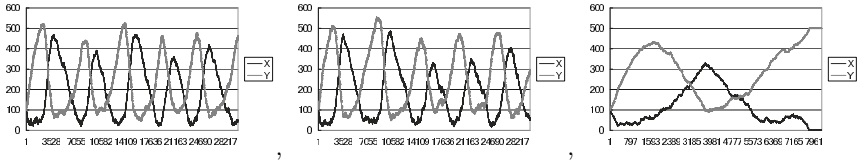


Fig. 4. Disappearance of periodic behavior of ARMS for the Brusselator system. In every simulation, parameters are $k_1 = 100$, $k_2 = 3$, $k_3 = 10^{-3}$, $k_4 = 1$ and the initial state is $(X, Y) = (100, 100)$. From left to right, the size of system is 1000, 530 and 200, respectively.

We investigated the behaviors of limit-cycle oscillations in the Brusselator with changing the number of molecules [13]. We defined the size of system s by $[x] + [y] \leq s$, where $[x]$ and $[y]$ denote the total number of each molecular. When the size reaches s, r_1 cannot be applied because the other rules do not change the total number of molecules. This s was only one parameter and all other parameters were fixed; $k_1 = 100, k_2 = 3, k_3 = 10^{-3}$, $k_4 = 1$ and the initial state was $(X, Y) = (100, 100)$. We changed $s = 1000$, 530 and 200.

When s was 1000, the time evolution showed the limit-cycle (left in the fig.4), which is in good agreement with the kinetics of the differential equation model despite appreciable fluctuations [13]. As the number of involved molecules decreased to 530, the fluctuations in the amplitude of oscillation increase and the kinetics began to differ from those of the differential equation; however the periodicity was maintained and the time evolution remains quasi-periodic (Fig.4 in the middle).

When $s = 250$, the magnitude of fluctuations grew large and the system was trapped in an unreactive state where $X = 0$ (Fig.4 in the right); however the time evolution remains quasi-periodic oscillation and a variety of periods in oscillations could not be observed. These results illustrates that there would be various types of the SNE existing and their behaviors would be related to its dynamical characteristics. The coarse grained phase spaces would show underlying mechanisms among them and it is our future work.

Acknowledgment. This research was partly supported by Information Initiative Center, Hokkaido University JAPAN. Visualization of the probabilistic field were designed by Kenji Tsunoda.

References

1. Dittrich, P., Ziegler, J., Banzhaf, W.: Artificial chemistries, a review. Artif. Life 7(3), 225–275 (2001)
2. Gillespie, D.T.: Exact Stochastic Simulation of Coupled Chemical Reactions. J. Phys. Chem. 81(25), 2340–2361 (1977)
3. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting systems. J. Chem. Phys. 115(4), 1716–1733 (2001)
4. Manca, V.: String rewriting and metabolism: a logical perspective, in Computing with Bio-Molecules. Theory and Experiments, pp. 36–60. Springer, Heidelberg (1998)
5. Nicolis, G., Prigogine, I.: Exploring Complexity, An Introduction. Freeman and Company, San Francisco (1989)
6. Hofbauer, J., Sigmund, K.: The Theory of Evolution And Dynamical Systems. Cambridge Univ. Press, Cambridge (1988)
7. Suzuki, Y., Tsumoto, S., Tanaka, H.: Analysis of Cycles in Symbolic Chemical System based on Abstract Rewriting System on Multisets. In: Artificial Life V, pp. 482–489. MIT Press, Cambridge (1996)
8. Suzuki, Y., Tanaka, H.: Chemical evolution among artificial proto-cells. In: Artificial Life VII, pp. 54–64. MIT Press, Cambridge (2000)
9. Suzuki, Y., Fujiwara, Y., Takabayashi, J., Tanaka, H.: Artificial Life Applications of a Class of P Systems: Abstract Rewriting System on Multisets. In: Calude, C.S., Pun, G., Rozenberg, G., Salomaa, A. (eds.) Multiset Processing. LNCS, vol. 2235, pp. 299–346. Springer, Heidelberg (2001)
10. Suzuki, Y., Davis, P., Tanaka, H.: Emergence of auto-catalytic structure in stochastic self-reinforcing reaction networks. J. Artif. and Robot. 7, 210–213 (2003)
11. Suzuki, Y., Tanaka, H.: Modeling P53 signaling network by using multiset processing. In: Applications of Membrane Computing, pp. 203–215. Springer, Tokyo (2006)
12. Suzuki, Y.: An investigation of the Brusselator on the mesoscopic scale. Inter. J. of Parallel, Emergent and Distributed Sys. 22, 91–102 (2007)
13. Umeki, M., Suzuki, Y.: A Simple Membrane Computing Method For Simulating Bio-chemical Reactions 27(3), 529–550 (2008)

Memory-Based Cognitive Framework: A Low-Level Association Approach to Cognitive Architectures

Paul Baxter and Will Browne

Cybernetics Intelligence Research Group, School of Systems Engineering,
University of Reading, Reading RG6 6AY, U.K.
{p.e.baxter,w.n.browne}@reading.ac.uk

Abstract. At its most fundamental, cognition as displayed by biological agents (such as humans) may be described as being the manipulation and utilisation of memory. A low-level approach to the associative sensory-motor development of cognition is then appropriate, rather than the more common higher-level functional approach. A novel theoretical framework – the memory-based cognitive framework (MBCF) – is proposed based upon these considerations. A computational architecture based on the MBCF is implemented on a mobile robot platform, and experimental results are presented to demonstrate the functionality of the architecture. It is shown that this low-level, bottom-up, approach can produce adaptive behaviours, which may ultimately form the foundation of cognitively flexible agents.

Keywords: memory-based cognition, cognitive robotics, autonomous mental development.

1 Introduction

The development of artificial agents with autonomous and adaptive behaviour is an ongoing goal for cognitive robotics research. Biological agents (most notably mammals) provide arguably the best examples of these properties, and so are the source of design concepts and principles [1]. One such fundamental principle obtained from biological theory is that cognition is fundamentally concerned with the manipulation and utilisation of memory [2].

Three further such principles are used as basic assumptions in the present work. Firstly, memory is essentially an associative process. Secondly, memory is representative of an agents history of interaction with its environment. Finally, there must be some form of innate mechanism that enables a means of functionally applying this memory to goal directed behaviour.

On this basis the novel Memory-Based Cognitive Framework (MBCF) for a mobile robot is proposed. This paper details the theory and an implementation of the MBCF [3]. It is proposed that the MBCF will result in an agent capable of autonomous and adaptive behaviour. There is a particular emphasis on allowing the specific functionality of the architecture to develop through interaction with the world, and not through an *a priori* or otherwise human-centred approach [4, 5].

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 402–409, 2011.

In the context of this paper, the term 'framework' (and hence the MBCF) refers to a descriptive theory, and 'architecture' is used to denote one computational implementation of this theory.

The remainder of this paper is organised into four main sections. In the first, the necessity for the proposed theoretical framework is examined, the fundamental considerations are outlined, and the framework itself presented. In the second part, the fundamental features of a computational implementation of this framework are presented. In the third part an experimental setup is described which will elucidate its functionality. Finally, the results are discussed in the context of applicability to the development of cognitive robotics, and further extensions to the architecture presented.

2 The Memory-Based Cognitive Framework

2.1 The Motivation for a New Framework

When applying biologically-based functional principles to robotics work, two broad approaches have been used: neural modelling, and behavioural modelling.

In the first, a model of neural connectivity serving the function of interest is implemented as an artificial neural network. This approach is essentially one of learn-by-building, where further understanding of the biological system of interest may be gained through the implementation of biologically-plausible neural mechanisms for a behavioural task (e.g. [6]). In the second approach, computational architectures are implemented based on higher level cognitive psychological theories and emphasises the behavioural functionality over the specific neural implementation (e.g. [7]).

Both of these approaches collectively stress the need for embodiment in the real world as a prerequisite for the emergence of intelligent behaviour through learning. However, this form of embodiment does not necessarily place the constraints on the computational architecture that the 'body' of a biological agent would place on its nervous system as it does not provide mutual constraints. Furthermore, in terms of autonomy of learning and behaviour, the high-level nature of information in the behavioural modelling approach proves problematic if the task environment changes significantly because of a lack of flexibility. Finally, it may be argued that the transparency (in terms of ability to explain the causes of produced behaviour) of neural system-based modelling approaches, such as that presented in the neural modelling approach, is reduced as the fidelity (and hence complexity) of the model increases. As a general observation, these approaches tend to focus on what may be described as higher-level cognitive functions.

The novel MBCF addresses these issues by incorporating two considerations. Firstly, the utility of neural systems-based architectures for the potential elucidation of biological mechanisms (at least in terms of function). Secondly, the theoretical and philosophical considerations of autonomy [8] and embodiment [9] that describes the conditions under which the biological agents have developed – an aspect frequently overlooked in current robotics work (with one other exception being the concept of homeokinesis [10]). By combining these two elements, the architecture may be used to explore issues which arise at the intersection of the two approaches discussed, and more theoretical considerations.

2.2 Neuroscientific Inspiration for the MBC Framework

Based on neurophysiological evidence, there are a number of models of the human cognitive architecture which attempt to provide an account of a wide range of cognitive functions. Indeed, models of working memory for example have generally endeavoured to do so. Of particular interest for the present work are the increasingly prevalent theories which emphasise the distributed nature of cognitive functionality (especially memory) over the competing modular view (e.g. [11, 12]). One such model is the "Network Memory" theory [2], which is a theory of human cortical and sub-cortical organisation and functioning. It is of particular applicability to the present work as it provides a wide ranging theory of human cognition. The MBCF takes inspiration from this theory in order to create a biologically non-implausible structure within which it can take shape.

There are three central ideas which underlie the Network Memory theory, which are of particular relevance to the present work:

- That memory is at its most basic an associative process.
- That memory is distributed across the brain: units of memory (called 'cognits' [13]) are distributed, overlapping networks of neurons which encode specific pieces of information (or indeed other cognits).
- That these distributed memories are informally arranged both heterarchically and hierarchically; with one of a dual hierarchy based in the motor regions, and the other in the perceptual regions.

2.3 The MBC Framework

The considerations discussed are brought together in the Memory Based Cognitive framework: figure 1 gives a functional overview. The sensory and motor spaces are defined by the particular embodiment of the agent. It is not just the number of sensors and their respective resolutions which are defined, but also the sensory and motor morphology of the agent implicitly provides constraints, resulting in the tight coupling of computational architecture and embodiment.

During the embodied agents' interaction with its environment, base elements in the sensory and motor spaces acquire activation. Equation (1) details the spread of activation: A_t is the activation of a cognit, i, at time t, α is a scaling factor for its previous time-step activation, and the second term is the summation of activation of all lower level cognits j which link to cognit i (with scaling factor β). Whilst α and β are necessary for the functioning of the computational architecture, they are not of theoretical importance to the MBCF, and hence are empirically tuned.

$$A_t^i = \alpha A_{t-1}^i + \beta \sum A_t^j \tag{1}$$

Upon this basis, associative links (which are named 'cognits' as inspired by the Network Memory theory) are formed reactively. These cognits may encode (or indeed may be said to represent) both spatial associations (i.e. co-occurrence on the same time-step), or temporal associations (occurrence on subsequent time-steps), and may in turn linked to one another. This process allows the development of an informal

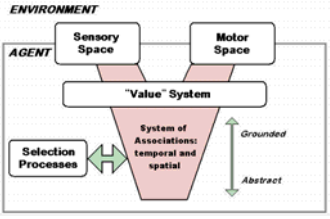


Fig. 1. Functional system overview of the MBCF. The sensory and motor spaces tightly integrate the system with its physical instantiation. The "value" system plays a central role in the development of the system and its resultant behaviour.

hierarchy structure of cognits. As this structure gains in complexity over time (i.e. as the agent interacts with its environment), the activation flow in the 'higher level' cognits will have the potential to have greater effect in biasing 'lower level' cognits, thereby allowing what may be described as top-down control to emerge.

3 Computational Implementation

A computational architecture has been derived from the framework proposed in the previous section: the cognit hierarchies, the value system, and the embodiment. This has resulted in Embodied MBCF Agents (EMA's), where each EMA is a system defined by the inextricable links between the specific hardware embodiment (i.e. mobile robot), and the computational architecture.

For the elucidation of basic operation, the computational architecture of the EMA implemented for the present study uses the functional structure shown in figure 2a, and divides the hierarchy into explicit levels. This type of discretisation of what is theoretically a continuous and informal hierarchy has been used in a computational model of the Network Memory theory, which explored high level behaviours [14]. The sensor and motor space sizes are defined by the mobile robot platform used; for example each possible sensor reading becomes a base element in the sensor space (see figures 2(b) and 2(c)), and is created upon first encounter. Sensory cognits in the sensory association layer are formed based on these base elements. This mechanism also applies to the other association layers.

The value system is necessary in this architecture and in the generation of behaviour as it biases the flow of activation in the cognit hierarchy, by means of a mechanism which may be seen as analogous to an emotion or homeostatic systems. The activation of sensory-motor cognits is scaled according to the value tag, which is derived from the parameters of the value system. Since it is the totality of activation in the motor cognits and motor space which determines the motor action executed, this process in effect provides the evaluation mechanism required for even this low-level goal-directed behaviour.

The concept of cognits are central to the computational architecture, although it should be noted that it is the functional role of cognits which are of interest (i.e. that they encode some associative relationship) rather than their proposed neural basis. Consequently, cognits are here represented by explicit encodings of an associative

relationship, as described in figure 2(b). As discussed previously, cognits are created retrospectively in response to the activation levels in other layers: whilst the creation of cognits on the sensory side is driven by incident sensory readings, motor cognits are initially formed by a random selection method (essentially a primitive form of motor babbling).

The EMA action selection scheme is not a centralised mechanism, which is contrary to many current computational cognitive architectures. As seen in figure 2(a), the biased flow of activation ends in the motor space at the end of each time-step, where the motor base element with the highest activation value gets executed (for each motor). Thus it is the totality of the activation flow throughout the hierarchy system which determines the next action of the EMA. This is an important point, as it means that with the continual changes in the numbers of cognits in each layer, the EMA does not simply build up a static state-action mapping (which, particularly in the case of sensory aliasing, may result in inappropriate actions [15]), but maintains behavioural flexibility.

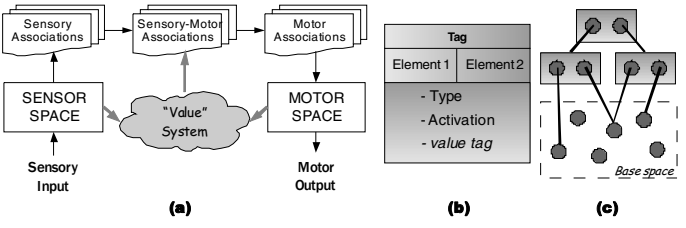


Fig. 2. (a) The layers of the EMA computational architecture implemented for this work. The link between the Sensory and Motor sides of the architecture is made through the Sensory-Motor association layer. The "Value" system modulates the activity of the Sensory-Motor layer. The arrows indicate the flow of activation over the course of one time-step. (b) A cognit: the tag is used for linking to other cognits, the tags of two lower level elements to be associated (and the type of association), and activation value. (c) The construction of cognits from either base elements or lower level cognits – these are spatial associations.

4 Experimental Setup and Results

4.1 EMA and Experimental Setup

The computational architecture described in the previous section was implemented in the object oriented programming language C#. The small mobile robot platform Miabot (Merlin Robotics) provides the embodiment: three ultrasonic sensors are used for the sensory input, and two wheel motors as effectors (figure 3).

Given the very simple sensory and motor capabilities of the resultant EMA, the target task for this study is for the agent to maximise both the distances detected by the sensors and the forward movement of the motors (i.e. an explore/object-avoid task). Of interest here is not just that the EMA achieves the task, but how this behaviour develops over time.

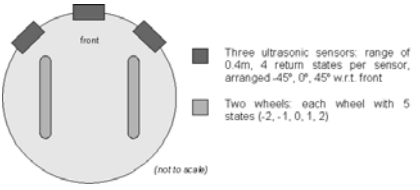


Fig. 3. Representation of the physical agent. This initial embodiment is very simplistic with three ultrasonic sensors, and two motorised wheels.

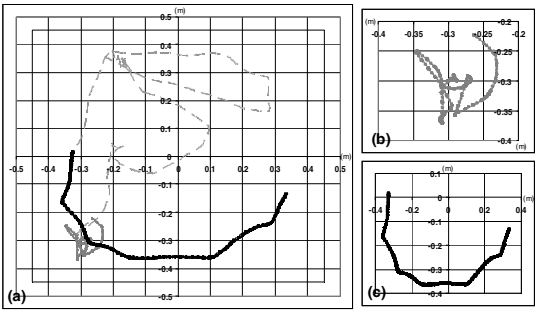


Fig. 4. (a) Sample path produced by the EMA in a square enclosed environment (boundary walls marked in black) over 500 time-steps – the starting position is (-0.3m, -0.3m). Two sub-sections of this path are shown: (b) the first 100 time-steps, and (c) the final 100 time-steps.

The parameters of the computational architecture and value system (found through empirical tuning) are set so that all cognit activation levels are reset at the end of each time-step ($\alpha = 0$). Additional cognits are created, where there is no duplication of existing cognits, on each time-step. This results in a system which may on any given single time-step be considered reactive, but which over time (as new cognits are created) can be described as being adaptive.

4.2 Results

In an open but bounded environment (figure 4a), runs of the EMA were compared with runs of a random walker controller (i.e. random motor action, with the same possible motor values as those in the EMA). With the number of runs n =10, the distance covered by the EMA was greater than that covered by the random walker, but not significantly (one-sided t-test: $p = 0.085$). This result may be explained by the fact that the motor space sizes are small, so the difference in total distance travelled would not be great. More interestingly, the area covered by the EMA in this environment was significantly greater than that covered by the random walker (one-sided t-test: $p = 0.001$) indicating that, as shown in figure 4 for example, the objective of the task is met.

Figure 4a shows a typical free-movement path taken by the EMA during a trial 500 time-steps long in a simulated environment. During the initial stages of the run, it can be seen (figure 4b) that the behaviour of the EMA approximates that of a random walker. However, by the last 100 time-steps of the simulation, when the agent is

moving in open space, the EMA only moves forward, turning away from obstacles (figure 4c). This sample run shows how the EMA starts with effectively random movements whilst the number of cognits increases (figure 4b), before developing behaviours which enable it to move forwards, but away from the walls which surround the otherwise open environment (figure 4c).

5 Discussion

Implementing the architecture in a mobile robot provides its embodiment, which provides a means of constraining the system by defining the sensory and motor spaces, and tying any subsequent development to the particular embodiment. This is important, as it results in all acquired 'information' by the agent being inherently based in, and with respect to, the agents' sensors and effectors, and not in some modality-free representation scheme. Another consequence of this is that the developed 'knowledge' is not in a form immediately amenable to human inspection: the aim of this system is not to provide a human readable map (or equivalent), but to allow the agent itself to interact with the architecture in a 'meaningful' way (where meaningful in this context implies some *a priori* human designer specifications - an issue which is to be addressed in future work). Enabling this explicit grounding of knowledge/information in the agents' sensory-motor morphology would also address any objections relating to the Symbol Grounding problem [16].

One of the novel aspects of the implemented computational architecture is that it represents associative links (i.e. cognits) as explicit constructs. This enables a level of interrogation into the development of the control system which would not be possible with an artificial neural network (ANN) approach. Additionally, this setup enables a higher level of computational flexibility since the number of connections between the cognit constructs are not static (as synapses are in most ANN's), and may change on the time-scale of single time-steps. One drawback to this approach however is that with increased sensory and motor space sizes, the combinatorial explosion in potential numbers of cognits may pose a problem from a computational load point of view. However, given the stated emphasis on using this framework and architecture as a research tool (and not as a directly applicable control system for industrial processes for example), it is held that this position in the trade-off is justified.

The version of the computational architecture presented in this paper only implements the basic aspects of the developed MBCF. Further work is ongoing to more completely represent the theoretical framework in the computational architecture. Three aspects are being focused on in particular: (1) a second level of the hierarchy (as described in figure 2(a)); (2) allowing cognits to maintain activation over subsequent time-steps, thus adding an additional temporal dimension to the generation of behaviour; and (3) exploring alternative implementations for the value system, with emphasis on adaptive biologically non-implausible structures.

6 Conclusion

This paper has shown that the low-level, associative-based computational architecture EMA can produce adaptive behaviours in a physically embodied agent. Even though

the behaviours developed are simplistic, it validates the proposed MBCF as a bottom-up approach to cognitive architectures.

However, a number of outstanding issues remain related to bringing the computational architecture closer to the MBCF theory. It remains an open question as to how far a fundamentally associative approach such as this can go towards high-level cognitive functionality – the present work has only started on this path.

References

1. Guillot, A., Meyer, J.-A.: The Animat contribution to Cognitive Systems Research. Journal of Cognitive Systems Research 2, 157–165 (2001)
2. Fuster, J.M.: Network Memory. Trends in Neuroscience 20, 451–459 (1997)
3. Baxter, P., Browne, W.: Towards a developmental memory-based and embodied cognitive architecture. In: Epigenetic Robotics 8. Lund University Cognitive Studies, University of Sussex (2008)
4. Berthouze, L., Metta, G.: Epigenetic robotics: modelling cognitive development in robotic systems. Cognitive Systems Research 6, 189–192 (2005)
5. Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M., Thelen, E.: Autonomous mental development by robots and animals. Science 291, 599–600 (2001)
6. Krichmar, J.L., Edelman, G.: Principles Underlying the Construction of Brain-Based Devices. Presented at AISB 2006, Bristol (2006)
7. Kawamura, K., Peters, R.A., Bodenheimer, R.E., Sarkar, N., Park, J., Clifton, C.A., Spratley, A.W., Hambuchen, K.A.: A parallel distributed cognitive control system for a humanoid robot. International Journal of Humanoid Robotics 1, 65–93 (2004)
8. Boden, M.: Autonomy: What is it? BioSystems 91, 305–308 (2008)
9. Ziemke, T.: What's that thing called Embodiment?. Presented at 25th Annual Meeting of the Cognitive Science Society (2002)
10. Hesse, F., Der, R., Herrmann, J.M.: Reflexes from self-organising control in autonomous robots. In: Epigenetic Robotics 7 (2007)
11. Postle, B.R.: Working memory as an emergent property of the mind and brain. Neuroscience 139, 23–38 (2006)
12. Chadderdon, G.L., Sporns, O.: A large-scale neurocomputational model of task-oriented behavior selection and working memory in prefrontal cortex. Journal of Cognitive Neuroscience 18, 242–257 (2006)
13. Fuster, J.M.: The Cognit: a network model of cortical representation. International Journal of Psychophysiology 60, 125–132 (2006)
14. Botvinick, M.M.: Multilevel structure in behaviour and in the brain: a model of Fuster's hierarchy. Philosophical Transactions of the Royal Society B 362(1485), 1615–1626 (2007)
15. Bagnall, A.J., Zatuchna, Z.V.: On the classification of maze problems. In: Bull, L., Kovacs, T. (eds.) Foundations of Learning Classifier Systems, pp. 307–316. Springer, Heidelberg (2005)
16. Harnad, S.: The symbol grounding problem. Physica D 42, 335–346 (1990)

Modelling Coordination of Learning Systems: A Reservoir Systems Approach to Dopamine Modulated Pavlovian Conditioning

Robert Lowe[1], Francesco Mannella[2], Tom Ziemke[1], and Gianluca Baldassarre[2]

[1] University of Skövde
Informatics Research Centre
Cognition & Interaction Lab
{robert.lowe,tom.ziemke}@his.se
[2] Consiglio Nazionale delle Ricerche
Istituto di Scienze e Tecnologie della Cognizione
Laboratory of Computational Embodied Neuroscience
{francesco.mannella,gianluca.baldassarre}@istc.cnr.it

Abstract. This paper presents a biologically constrained reward prediction model capable of learning cue-outcome associations involving temporally distant stimuli without using the commonly used temporal difference model. The model incorporates a novel use of an adapted echo state network to substitute the biologically implausible delay chains usually used, in relation to dopamine phenomena, for tackling temporally structured stimuli. Moreover, the model is based on a novel algorithm which successfully coordinates two sub systems: one providing Pavlovian conditioning, one providing timely inhibition of dopamine responses to salient anticipated stimuli. The model is validated against the typical profile of phasic dopamine in first and second order Pavlovian conditioning. The model is relevant not only to explaining the mechanisms underlying the biological regulation of dopamine signals, but also for applications in autonomous robotics involving reinforcement-based learning.

Keywords: reward prediction, reservoir dynamics, dopamine, Pavlovian conditioning, reinforcement learning.

1 Introduction

The learning processes underlying Pavlovian conditioning have been related to the phasic dynamics of dopamine (DA), a neuromodulator produced within brain areas such as the ventral tegmental area (VTA) and shown to play a fundamental role as a signal in trial-and-error learning processes based on reinforcement [1]. DA dynamics have been mathematically captured by the seminal Rescorla-Wagner model [2] which proposed that learning is driven by discrepancies (errors) between *actual* and *anticipated* rewards. This model has since been generalized to conditioned stimuli (CS) and unconditioned stimuli (US) which are experienced at different times and can involve multiple rewards through the proposal of algorithms based on the *temporal-difference reward-prediction error* rule [3]

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 410–417, 2011.

('TD rule'). This model is based on the formation of a prediction, *at each time step*, of the expected sum of the future discounted rewards. Notwithstanding its theoretical importance, the TD rule suffers from limitations [4,5, cf.]. One is that it produces a progressive retro-propagation in time of the DA signal, from the US to the CS, which has not been observed in animals.

Models have recently been proposed to overcome such limits. Alexander and Sporns [6] have proposed an embodied model focused on both classical and instrumental learning and based on two sub-systems, one for producing DA phasic responses (at US and CS onset), and one for the timely learned inhibition of them (e.g. at US onset after learning). The model is capable of reproducing important empirical data, e.g. acquisition (transfer of the DA signal from US to CS onset), US omission (dopamine dip when an expected US is omitted), and extinction (dopamine returns to baseline when the CS ceases to be followed by the US). The model, however, is rather abstract, mapped onto real brain anatomy only at a shallow level. Some of its internal connections are also rather ad hoc, in particular within the sophisticated VTA micro-anatomy. Furthermore, it relies on a 'non-biological' mechanism to implement the timely inhibition of the DA burst at US onset, i.e. the 'delay chain', often exploited within the DA modeling literature. Solutions based on delay chains have a strong limit of scalability. In fact, if they are used to process multiple temporally overlapping stimuli they have difficulties in tackling concomitant non-linearly separable problems thereby requiring an increasing number of hand-crafted delay chains to solve them.

A second important model [7], mainly focused on Pavlovian learning, is again based on two sub-systems, one for production of phasic responses and one for their timely inhibition. The model can reproduce the outcomes of various classical conditioning experiments and is closely mapped onto brain anatomy. However, it is also limited in that it relies upon some mechanisms which are implemented in abstract/non-neural terms, again it relies upon a delay chain mechanism, and finally it does not propose a detailed account of how the learning processes taking place in the two sub-systems can be coordinated in time. The latter point is particularly important because if the two learning processes are not coordinated, the faster convergence of one of the two might prevent the convergence of the other, for example if the learning of the timely inhibition of DA at the US onset is too fast, it might prevent learning of the anticipatory DA burst at CS onset.

The model that we propose here is a first step towards the solution of such limits. First, it is implemented in fully neural network terms. Second, it implements the Pavlovian sub-system causing the CS anticipatory DA burst based on a biologically plausible model of the amygdala drawn from [5]. Third, it substitutes the rather artificial delay chain mechanism with a biologically plausible neural version of a reservoir system [8,9]. Finally, it proposes detailed mechanisms to implement the coordination between the learning process producing the CS-based anticipatory DA response and the learning process causing the DA timely inhibition at US onset.

The rest of the paper is organized as follows. Section 2 presents the architecture and functioning of the model. Section 3 reports the functioning of the model in first and second order conditioning tests. Finally, section 4 draws conclusions.

2 Methods

A potential solution to the delay chain problem mentioned in Section 1 can be found using a more biologically constrained approach. Taking inspiration from stimulus-induced activity in sensory cortex, reservoir systems [8,9] are a class of neural networks that have been demonstrated to be particularly adept at processing temporal stimuli e.g., see [10]. Two types of such networks – 'Liquid State Machines' (LSM) [8], and 'Echo State Networks' (ESN) [9,11] – have demonstrated how spatial-temporal input signals to the network (reservoir) can be accurately replicated, through the use of linear units that read out the reservoir node activity, even after significant delays between stimulus input and reservoir output providing a short term memory mechanism [9]. This mechanism has biological plausibility: "[I]t is plausible that a biological read-out neuron can learn to decode the active states of a recurrent network through trial and error in a reward-based setting" [10, p.117]. The model presented here deploys an ESN for processing stimulus inputs in discrete time in a manner that departs from the classical temporal difference model. Figure 1 schematizes the architecture of the model. In the model (Figure 1), prefrontal cortex (PFC), amygdala (AMG), and ventral tegmental area (VTA) capture the basic functionality of their real biological counterparts with respect to reward prediction learning. The PFC, represented by an ESN, consists of $\mathbb{N}$ (= 40) synchronously updated sigmoid neurons. We follow the methodological approach of Jaeger [11] for imbuing the network with the echo state property: (a) use of sparse reservoir connectivity: 75% of reservoir weight matrix (W) set to zero, 25% set randomly in [-1,1]; (b) use of spectral radius and alpha scaling (in our experiments $\alpha = 0.95$). Furthermore, the vectors W_{CS1} and W_{CS2} - conditioned stimuli (CS) input weights to PFC - also have 75% of values set to zero with the remainder randomly set to values in [0,1] (mimicking positive glutamate inputs in pyramidal cells). Two novel features are introduced with respect to standard ESN to improve biological plausibility: (a) the use of only non-negative activation values in the reservoir:

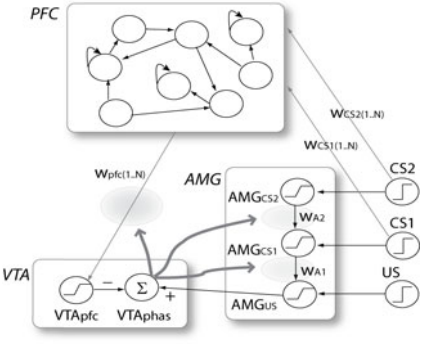


Fig. 1. Architecture of the model, mainly based on an ESN (PFC), a Hebbian associative network (AMG), and a simple network to produce phasic DA (VTA)

this was also done as negative activation interfered with the system's ability to perform extinction of PFC-VTA weights (interestingly, this change did not prevent pattern discrimination and learning); (b) the use of a DA-based online learning algorithm: online algorithms have been applied to ESN in cognitive science and robotics with promising results [12].

Inputs from AMG and PFC allow VTA to output negative or positive phasic DA signals (negative DA is meant to represent DA below baseline levels). AMG_{US} unit projection to VTA causes DA bursts which promote strengthening of PFC-VTA and intra-AMG connections. PFC input to VTA via weights W_{pfc} causes either a timely inhibition of DA burst, at US onset, caused by the CS (so the DA signal is near zero), or, when the US is omitted, a negative DA signal which leads to a weakening of PFC-VTA and intra-AMG connections. Note that in all equations presented below these notations are used:

$$f[x]{=}0 \; if \; x{<}0, \quad f[x]{=}x \; if \; 0{\leq}x{<}1, \quad f[x]{=}1 \; if \; 1{\leq}x \tag{1}$$

$$g[x]{=}{-}1 \; if \; x{<}{-}1, \quad g[x]{=}x \; if \; {-}1{\leq}x{<}1, \quad g[x]{=}1 \; if \; 1{\leq}x \tag{2}$$

AMG consists of three units which represent populations of neurons sensitive to two different CS and a US. As with VTA and PFC, AMG is activated in discrete time and provides a simplified version of the model of [5] that uses leaky neurons and a differential Hebbian learning rule. AMG, PFC and VTA units are updated as follows:

$$At\ onset{:}\ CS2{=}1, CS1{=}1, US{=}1 \quad else{:}\ CS2{=}0, CS1{=}0, US{=}0 \tag{3}$$

$$AMG_{\mathrm{CS2}}(t){=}f[CS2(t)] \quad AMG_{\mathrm{CS1}}(t){=}f[CS1(t){+}W_{\mathrm{A2}}(t){\cdot}AMG_{\mathrm{CS2}}(t))] \tag{4}$$

$$AMG_{\mathrm{US}}(t){=}f[(US(t){+}W_{\mathrm{A1}}{\cdot}AMG_{\mathrm{CS1}}(t))] \tag{5}$$

$$PFC_i(t){=}max[0, tanh[\textstyle\sum_{j\in\mathbb{N}}[W_{ij}{\cdot}PFC_j(t{-}1)]{+}W_{\mathrm{CS1}}i{\cdot}CS1(t){+}W_{\mathrm{CS2}}i{\cdot}CS2(t)]] \tag{6}$$

$$VTA_{\mathrm{pfc}}(t){=}f[\textstyle\sum_{i\in\mathbb{N}}[W_{\mathrm{pfc\ i}}(t{-}1){\cdot}PFC_i(t)]] \quad VTA_{\mathrm{phas}}(t){=}g[AMG_{\mathrm{US}}(t){-}VTA_{\mathrm{pfc}}(t)] \tag{7}$$

AMG connection weights are updated using eligibility traces (E_{CS1} and E_{CS2}) that abstract a continuous-time differential Hebbian learning rule [5, cf.] for which learning happens only if pre and post activation time difference (e.g., between CS1 and US, or CS2 and CS1) is within a certain time window:

$$E_{CS1}(t){=}max[CS1(t), \Omega{\cdot}E_{CS1}(t{-}1)] \quad E_{CS2}(t){=}max[CS2(t), \Omega{\cdot}E_{CS2}(t{-}1)] \tag{8}$$

$$W_{\mathrm{A1}}(t){=}\begin{cases} f[W_{\mathrm{A1}}(t{-}1){+}\eta{\cdot}VTA_{\mathrm{phas}}(t){\cdot}AMG_{\mathrm{US}}(t){\cdot}E_{CS1}(t)], & if \;\; VTA_{\mathrm{phas}}(t){>}0 \\ f[W_{\mathrm{A1}}(t{-}1){+}\eta{\cdot}VTA_{\mathrm{phas}}(t){\cdot}(US(t){=}{=}0){\cdot}E_{CS1}(t)], & if \;\; VTA_{\mathrm{phas}}(t){<}0 \end{cases} \tag{9}$$

$$W_{\mathrm{A2}}(t)=\begin{cases} f[W_{\mathrm{A2}}(t-1)+\eta\cdot VTA_{\mathrm{phas}}(t)\cdot AMG_{\mathrm{CS1}}(t)\cdot E_{CS2}(t)], & if\ \ VTA_{\mathrm{phas}}(t)>0 \\ f[W_{\mathrm{A2}}(t-1)+\eta\cdot VTA_{\mathrm{phas}}(t)\cdot (CS1(t)==0)\cdot E_{CS2}(t)], & if\ \ VTA_{\mathrm{phas}}(t)<0 \end{cases} \tag{10}$$

where Ω (= 0.9) is a decay constant, η (= 0.075) is a learning rate, and $(US(t) == 0)$ and $(CS1(t) == 0)$ are Boolean variables equal to 1 when US or $CS1$ are respectively equal to 0, and to 1 otherwise. Note that these Boolean variables are equal to 1 when the expected stimulus, either US or CS1, is missing: a more detailed model should involve a separate system for actively suppressing AMG stimuli representations. These learning rules imply that positive DA ($VTA_{phas} > 0$) augments intra-AMG weight values, whereas a negative DA ($VTA_{phas} < 0$) decrements such weight values (extinction; negative DA is induced by PFC in the absence of the post-synaptic stimulus, either US or CS1, at the step at which PFC-VTA weights have previously been strengthened).

The weights of PFC-VTA connections are updated using a different DA-based Hebbian learning rule and an eligibility trace (E_{pfc}) of the previous DA burst:

$$E_{pfc}(t)=max[VTA_{\mathrm{phas}}(t), \Omega\cdot E_{pfc}(t-1)] \tag{11}$$

$$W_{\mathrm{pfci}}(t)=\begin{cases} f[W_{\mathrm{pfci}}(t-1)+\kappa\cdot VTA_{\mathrm{phas}}(t)\cdot E_{pfc}(t-1)\cdot PFC_i(t)], & if\ \ VTA_{\mathrm{phas}}>0 \\ f[W_{\mathrm{pfci}}(t-1)+\kappa\cdot VTA_{\mathrm{phas}}(t)\cdot PFC_i(t)], & if\ \ VTA_{\mathrm{phas}}<0 \end{cases} \tag{12}$$

where Ω (= 0.9) is a decay constant, and κ (= 0.1) a learning rate.

Importantly, the effect of this rule is that learning takes place in PFC *only if the current DA burst* (VTA_{phas}), caused by AMG, *has been preceded by another dopaminergic burst* (E_{pfc}), caused by a previous activation of AMG. This is the core mechanism which allows the coordination between learning of AMG and PFC. In fact: (a) the formation of PFC-VTA weights can take place (and produce a timely inhibition of DA at US onset, or, in second order conditioning, at CS1 onset) *only after* the weights of AMG have started to form (to produce a DA burst at CS1 onset, or, in second order conditioning, at CS2 onset); (b) PFC connection weights cannot form in correspondence with a CS which is *the first predictor* of a subsequent CS-based or US-based DA burst: earlier formation of these weights would inhibit the DA induced by AMG in correspondence with the first 'unpredicted predictor' (either CS1 or, in second order conditioning, CS2); (c) in the case of extinction, PFC-VTA connection weights can decrease only when the omission of US, together with the PFC timely inhibition, induces a negative DA value (or, in the case of second order conditioning, only when the CS1-US AMG weight has decreased and the related DA signal at CS1 onset has become lower than the PFC inhibitory signal at CS1 onset): this implements a second form of coordination of the two learning processes where PFC follows AMG extinction step by step and cannot precede and prevent it.

3 Results

The model functioning was evaluated with a first order and a second order conditioning test. First order conditioning lasted 300 trials, each trial lasting 8

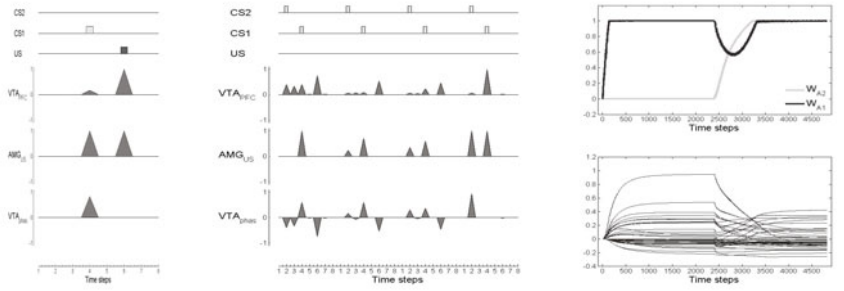


Fig. 2. Left: First order conditioning. At trial 300 AMG has learned to transfer the DA burst from US to CS1 onset and PFC to inhibit the DA burst at US onset. Middle: Second order conditioning. The four trials (301, 321, 341, 599, respectively) illustrate the gradual transfer of DA burst from CS1 to CS2 onset. Right top: Development of AMG weights over trials. Right bottom: Development of PFC-VTA weights.

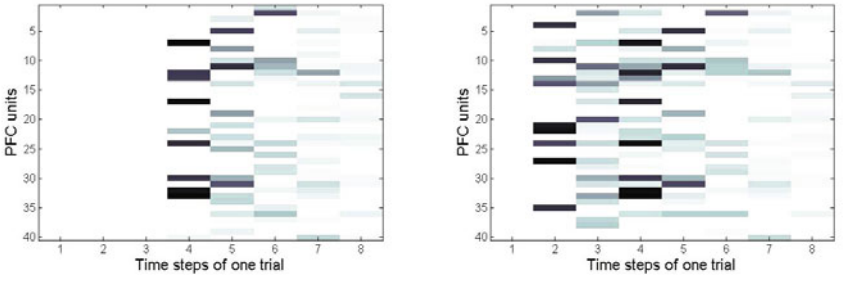


Fig. 3. Left: Reservoir activity in trial 300 having only the CS1 onset. Right: Reservoir activity in trial 301 having both the CS1 and CS2 onset.

steps with CS1 and US presented respectively at steps 4 and 6. Second order conditioning lasted a further 300 trials, composed of alternated first order trials (the 'reminder', run as in the previous phase) and second order trials (the actual second order conditioning: each of these trials lasted 8 steps, was run without US, and entailed the presentation of CS2 and CS1 at steps 2 and 4, respectively).

Figure 2 provides data from the two tests. The VTA_{phas} unit learns to produce an activation pattern characteristic of phasic dopamine [13]. Initially VTA_{phas} produces a phasic burst only at US onset (acquisition phase). With learning, VTA_{phas} progressively attenuates firing at US onset and 'transfers it' to CS1 onset ('conditioning': see left graphic). Presentations of 'CS2' followed by 'CS1' from trial 301 onwards leads to a progressive transfer of DA phasic response to 'CS2' (second order conditioning). This effect can be seen in the middle graph depicting trial number 301, 321, 341 and 599, respectively. The left-most middle graphic illustrates the first trial of second order conditioning. Note, in the absence of the now anticipated US a negative reward prediction error occurs subsequently weakening AMG (W_{A1}) and PFC-VTA weights (for this reason

CS2-CS1, CS1-US pairings are alternated to prevent extinction[1]). After the initial burst of VTA_{pfc} activity from 'CS2' to 'CS1' onset this DA error leads to gradual weakening of PFC weighted output to VTA_{pfc} (second left-most). In succeeding trials the DA error attenuates and VTA_{pfc} activity increases at 'CS1' onset (providing the 2nd DA burst following 'CS2') in accordance with the two-process coordinated learning algorithm described in section 2. A subset of PFC-VTA weights effectively replaces the weakened PFC-VTA weights over subsequent trials (see lower right graphic). This occurs as PFC-VTA weights at 'CS1' onset increase after, step by step, W_{A2}, connecting 'CS2' and 'CS1' in AMG, has started to form (upper right graphic).

Figure 3 shows an example of reservoir activity with only CS1 (left graph) and with both CS1 and CS2 (right graph). Note how, after CS2 onset, the temporally overlapping effects of CS1 and CS2 lead to produce similar, but distinct, patterns of activation at CS1 onset with respect to those produced by CS1 alone. This provides a means by which PFC can discriminate between different contexts and to associate to them the desired timely inhibition of DA. This simple demonstration, in line with [10], shows that reservoir systems have the power of distinguishing temporal input patterns through distributed activations.

4 Conclusions

This paper describes a model of Pavlovian conditioning exhibiting the ability to generate the typical profile of phasic dopaminergic activity observed in animal experiments on first and second order Pavlovian conditioning. The model is based on two complementary learning systems: (a) a first system, based on a discrete-time differential Hebb associative network putatively corresponding to the amygdala, which implements the core stimuli associations underlying Pavlovian conditioning; (b) a second system, based on a dynamical reservoir network and putatively representing some of the dynamical processes of prefrontal cortex, which implements timely inhibition of dopaminergic phasic responses caused by biologically salient *anticipated* stimuli.

From a neuroscientific perspective, the novelty of the model resides not only in its systemic and biologically plausible account of dopamine-related Pavlovian phenomena, which go beyond the temporal difference model, but also (a) in its use of a biologically plausible reservoir network to substitute the delay chain mechanism, commonly used to implement the timely inhibition of dopamine, and (b) in the proposal of specific learning mechanisms which allow the coordination of the learning processes related to the two systems composing the model.

The model can also be used as a template for designing autonomous robot neurocontrollers, in particular involving motivation-based behaviours and dynamic environments [14]. The integration of sensorimotor activity with non-neural bodily activity over time is of relevance to the understanding of motivational processes that achieve high level goals with flexible utilization of low

[1] Repeated omissions of the US per trial weakens PFC-VTA and AMG weights inducing gradual extinction of US and CS onset phasic responses (data not reported).

level behavioural and bioregulatory primitives [14]. A number of disembodied neural computational models of reward-based learning exist, e.g. [15], but only a small number of biologically relevant models of reward learning have been applied to autonomous robotics e.g. [6,16]. In this respect, distributed activation patterns of reservoir systems could support spatially and temporally flexible and robust learning in robotic systems interacting with noisy environments.

Acknowledgements. This work has been supported by a European Commission grant to the FP6 project "Integrating Cognition, Emotion and Autonomy" (ICEA, IST-027819, www.iceaproject.eu).

References

1. Schultz, W., Dayan, P., Montague, P.R.: A neural substrate of prediction and reward. Science 275, 1593–1599 (1997)
2. Rescorla, R.A., Wagner, A.W.: A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In: Black, A.H., Prokasy, W.F. (eds.) Classical Conditioning II: Current Research and Theory, pp. 64–99. Appleton-Century-Crofts, New York (1972)
3. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, Cambridge (1998)
4. Redgrave, P., Prescott, T.J., Gurney, K.: The basal ganglia: a vertebrate solution to the selection problem? Neuroscience 89, 1009–1023 (1999)
5. Mannella, F., Mirolli, M., Baldassarre, G.: The role of amygdala in devaluation: a model tested with a simulated robot. In: Berthouze, L., Prince, C.G., Littman, M., Kozima, K., Balkenius, C. (eds.) Proc. 7th Int. Conf. on Epigenetic Robotics, pp. 77–84. Lund University Cognitive Studies (2007)
6. Alexander, W.H., Sporns, O.: An Embodied Model of Learning Plasticity, and Reward. Adaptive Behavior 3-4, 143–159 (2002)
7. O'Reilly, R.C., Frank, M.J.: PVLV: The Primary Value and Learned Value Pavlovian Learning Algorithm. Behavioral Neuroscience 121, 31–49 (2007)
8. Maass, W., Natschlager, T., Markram, H.: Real-time computing without stable states: a new framework for neural computation based on perturbations. Neural Computation 14, 2531–2560 (2002)
9. Jaeger, H.: Short term memory in echo state networks. GMD Report 152 (2001)
10. Buonomano, D.V., Maass, W.: State-dependent computations: spatiotemporal processing in cortical networks. Nat. Rev. Neurosci. 10, 113–125 (2009)
11. Jaeger, H.: A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach. GMD Report 159 (2005)
12. Hertzberg, J., Jaeger, H., Schoenherr, F.: Learning to Ground Fact Symbols in Behavior-Based Robots. In: van Harmelen, F. (ed.) Proc. 15th European Conf. on Artificial Intelligence, pp. 593–600. IOS Press, Amsterdam (2002)
13. Suri, R.E.: TD models of reward predictive responses in dopamine neurons. Neural Networks 15, 523–533 (2002)
14. Ziemke, T., Lowe, R.J.: On the Role of Emotion in Embodied Cognitive Architectures: From Organisms to Robots. Cognitive Computation 1(1), 104–117 (2009)
15. Lowe, R.J., Humphries, M., Ziemke, T.: The dual-route hypothesis: evaluating a neurocomputational model of fear conditioning in rats. Connection Science 21(1), 15–37 (2009)
16. Mannella, F., Baldassarre, G.: A Neural-Network Reinforcement-Learning Model of Domestic Chicks that Learn to Localise the Centre of Closed Arenas. Phil. Trans. R. Soc. B. 362, 383–401 (2006)

Evolving Central Pattern Generators with Varying Number of Neurons

Jeisung Lee and DaeEun Kim

Biological Cybernetics Lab
School of Electrical and Electronic Engineering
Yonsei University
Shinchon, Seoul, 120-749, Corea, South Korea
{leejaisung,daeeun}@yonsei.ac.kr

Abstract. Central pattern generator (CPG) is a kind of neural circuit which can be observed in many animals showing rhythmic patterns of actions. The CPG neural circuit can produce complex rhythmic patterns by receiving only simple signals from the brain. Generally, the CPG neural models can be applied to solve robotic problems, or to understand the underlying neural mechanism for rhythmic animal behaviours. In this paper, we focus on how a small number of neurons generate the variable frequencies and phase of motor actions, and inspect what is the capacity of a varying number of neurons as a CPG model. The performance measure consists of frequency variability, input/output response rate, and phase shift. We have used evolutionary computation to measure the best performance for each number of CPG neurons ranging from two to eight neurons, and the result shows that four neurons or more can easily generate variable frequencies and anti-phase difference for left and right motor actions.

Keywords: central pattern generator, locomotion, frequency and phase.

1 Introduction

Neural networks in the spinal cord have been shown to produce rhythmic patterns. These neural circuits are called central pattern generator (CPG) [1]. The CPG is able to generate complex rhythmic patterns with simple signal inputs from the brain. It is possible to simulate the rhythmic movement of the animals, when the rhythmic patterns are applied to their motor neurons. Locomotion problem in robotics is often handled with the CPG neurons. For example, biped working robot [2,3], quadruped robot [4], hexapod robot [5], and lamprey [6,7] have been developed with the neuron model. Recently Ijspeert et al. [7] have developed the salamander robot that has a similar behavior pattern found in the real salamander animal [8]. The neuron model follows the CPG structure in the real salamander.

Lewis et al. [2] have applied evolutionary computation to build a CPG neuron model for a hexapod robot, where each leg has two neurons to generate

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 418–425, 2011.

oscillations and the CPG neurons have a phase difference of 90 degrees. After evolving the oscillation of each leg, six CPG neurons have been developed for effective walking behaviors. Billard and Ijspeert [4] used leaky-integrators with six neurons with symmetric structure, and applied it to the walking behavior of a AIBO robot. Ekeberg [6] showed a biologically plausible model of CPG neuron network. This CPG includes eight neurons with symmetric structure. The neurons have a specific role such as excitatory interneurons, contralateral inhibitory interneurons, lateral inhibitory interneurons, and motoneurons. The CPG model can change the frequency and magnitude depending on the input signal as well as maintain anti-phase for left and right motor signals.

The CPG model needs to show variable frequencies and phase shift for left and motor signals. The input signal is given from the brain system to control the frequencies and amplitudes of the motor action. Many researchers have shown a possible CPG model to demonstrate rhythmic motor actions. However, there has been no research for the capacity of CPG characteristics depending on the number of neurons. In this paper, we focus on the questions of how we can implement a simple structure of CPG neurons, how many neurons are needed to build an anti-phase shift for left and right motors, or how many neurons should be required to show variable frequencies depending on input signals. With genetic algorithm, we evolve a set of neurons to build the rhythmic pattern of motor actions, and we test a varying number of neurons from two neurons to eight, to check the performance of variable frequencies, input/output response and the phase difference between the two motors.

2 Simulation Method

2.1 Characteristics of CPG Neurons

We use genetic algorithms to evolve a group of neurons for CPG characteristics, frequency and phase of motor signals. The evolutionary computation can find appropriate neural parameters for desired oscillations. It is not a difficult problem to generate simple oscillations with a group of neurons. However, for general CPG model, the output neurons vary frequencies and amplitudes depending on the input. Several assumptions are needed to build such an oscillatory neuron model.

- Oscillations should have a fixed period for the same input.
- Left and right motors should have a fixed phase shift. Anti-phase are preferred.
- The frequency of oscillations should change depending on the input signal from the brain system. It is involved with speed of movement for locomotive actions.
- The magnitude of the two motor actions, left and right should change depending on the input. It is involved with power of motor muscles.
- The left and right motor neurons should have the same or similar output signals for the same input. This can be a condition to represent similar movement in the left and right of the robot.

To obtain the same level of motor outputs at the left and right side, we apply a symmetric structure with neural parameters including neuron weights to the CPG model. The CPG model consists of a group of neurons ranging from two neurons to eight. We investigate the CPG characteristics mentioned above for each number of neurons and thus determine the capacity or limitation of the CPG neurons depending on the number of neurons. That is, we can see how many neurons are required to generate general CPG characteristics, variable frequencies and phase difference between the left and right motor signals.

2.2 Evolutionary Computation Method

We use a leaky-integration neuron model with n neurons to evolve oscillators as follows:

$$\frac{dy_i}{dt} = \frac{1}{\tau}(-y_i + g \cdot u + \sum_{j=1}^{n} z(y_j)w_{ji}) + \eta \tag{1}$$

$$z(y_i) = 1/(1 + e^{\lambda(-y_i+p)}) \tag{2}$$

where w_{ji} is the weight from the j-th neuron to the i-th neuron, z is the sigmoidal function, τ is the decay constant, g is the gain factor, u is the input, λ is the scaling parameter, p is the bias term, and η is random noise.

Table 1. GA parameters for evolving oscillators

Population size	100
Generations	5000
Weight bounds	[-13, 13]
Crossover probability	0.6
Mutation probability	0.4
p	[-8, -4]
λ	[-2, 4]
τ	3
g	[-3, 10]

For genetic algorithms, the chromosome consists of n^2 weight connections, n time constants, n scaling terms, n gain factors, and n bias terms for n neurons. The evolutionary computation starts with 100 random chromosomes. New chromosomes are reproduced by recombination, and by the elitism strategy, the best chromosome is inserted into a new population. We evolved the neural parameters for CTRNN (Continuous Time Recurrent Neural Networks) [9] with multiple neurons ranging from two to eight, independently. For a given number of neurons, the evolutionary experiments were repeated ten times. We measured the performance with CPG characteristics.

2.3 Fitness Function

The genetic algorithm evolution is progressed in the following order:

1. Check if the neural network shows regular oscillations.
2. if the *oscillation fitness* F_1 is above a threshold of 0.7, check the phase difference between the two motor outputs. The oscillation fitness is measured by comparing the maximum / minimum peak amplitude with a given threshold value for a set of inputs.
3. if the *phase fitness* F_2 is more than 0.7, then compare the input/output response. The phase fitness is measured by the phase difference between the left and right motor actions.
4. if the *response fitness* F_3 is more than 0.7, then compare the frequency variability F_4. The response performance is measured with the proportion of the output change over the input change.
5. Finally, check the *frequency fitness* F_4, that is, relation between the input signal change and variability of oscillation frequencies. The larger range of frequencies can be generated, the higher fitness score is given.

The fitness function is as follows.

$$fitness = (F_1 + F_2 + F_3 + F_4)/4 \tag{3}$$

where F_1 is the oscillaton fitness, F_2 is the phase fitess, F_3 is the response fitness, and F_4 is the frequency fitness. We assumed the desired frequencies range from 0 Hz to 15 Hz. Normally, once oscillations are generated, the corresponding frequency can be calculated with the following equation:

$$y_n = \frac{1}{m} \sum_{k=1}^{m} x(k) e^{-j2\pi nk/m} \tag{4}$$

where m is the number of sample points in time, $x(k)$ is the k-th sample data point, n is the frequency value, and y_n is the complex value with real and imaginary part for the n-th frequency. The absolute value $|y_n|$ determines the magnitude of the n-th frequency. Also, the phase value for a given time sequence of signals can be obtained with the equation:

$$\phi = \text{atan2}(\text{real}(y_n),\ \text{imag}(y_n)) \tag{5}$$

The frequency fitness F_4 is measured by the difference between the maximum frequency and the minimum frequency generated for given input signals. The phase difference F_2 is calculated as the difference of phases calculated for the left and right motor signals, respectively. Each F_k for $k = 1, .., 4$ is normalized in the range [0, 1]. $F_k = 1$ indicates the best score for each of CPG characteristics.

If more variety of frequencies can be generated with input signals, we can obtain a larger range of movement speeds. The input/output response rate F_3 can show the efficiency to control the motor actions. The phase difference between the two motor signals can have maximum 180 degrees. The locomotive pattern often shows the anti-phase for left and right motor signals.

3 Experiments

We test the CPG models for a varying number of neurons–see Fig.1. Three neurons or less with the symmetric structure cannot generate the anti-phase

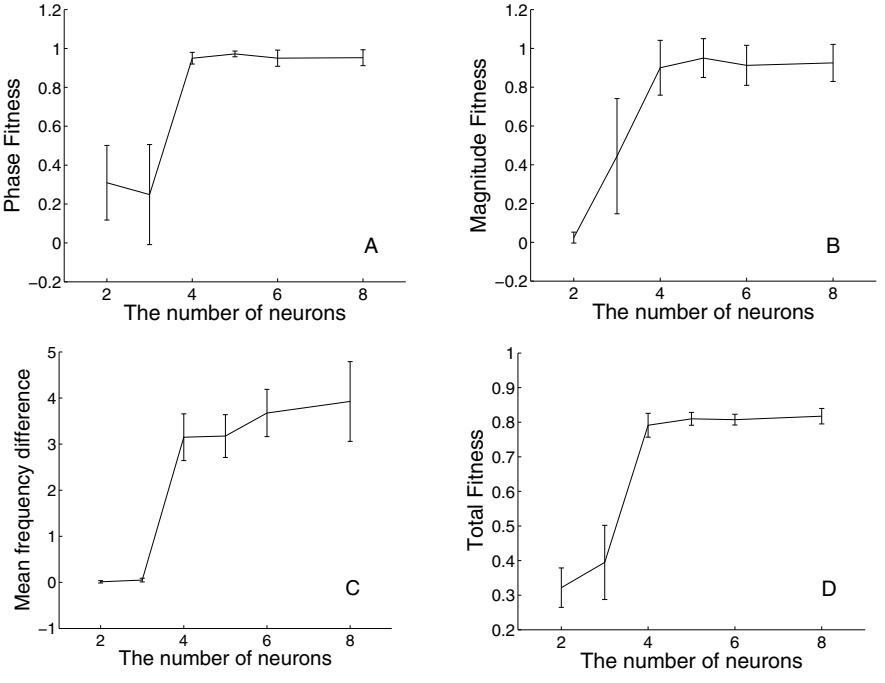


Fig. 1. Fitness components depending on the number of neurons (A: phase fitness with the number of neurons, B: magnitude fitness, C: mean frequency difference, D: total fitness)

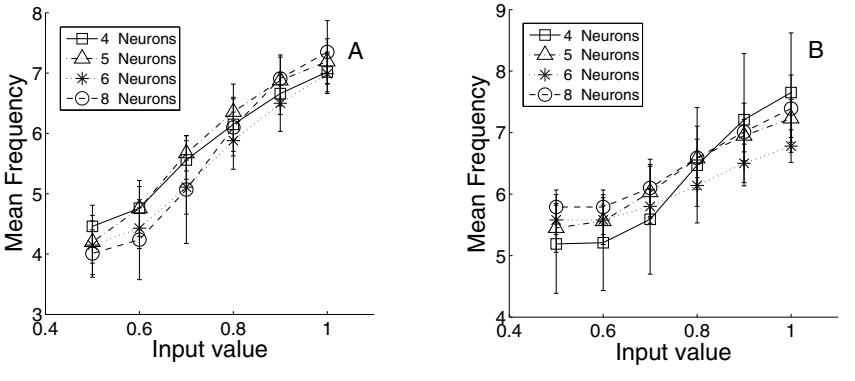


Fig. 2. Variable frequencies depending on the number of neurons (left: the same input signals are given, right: different input signals are given)

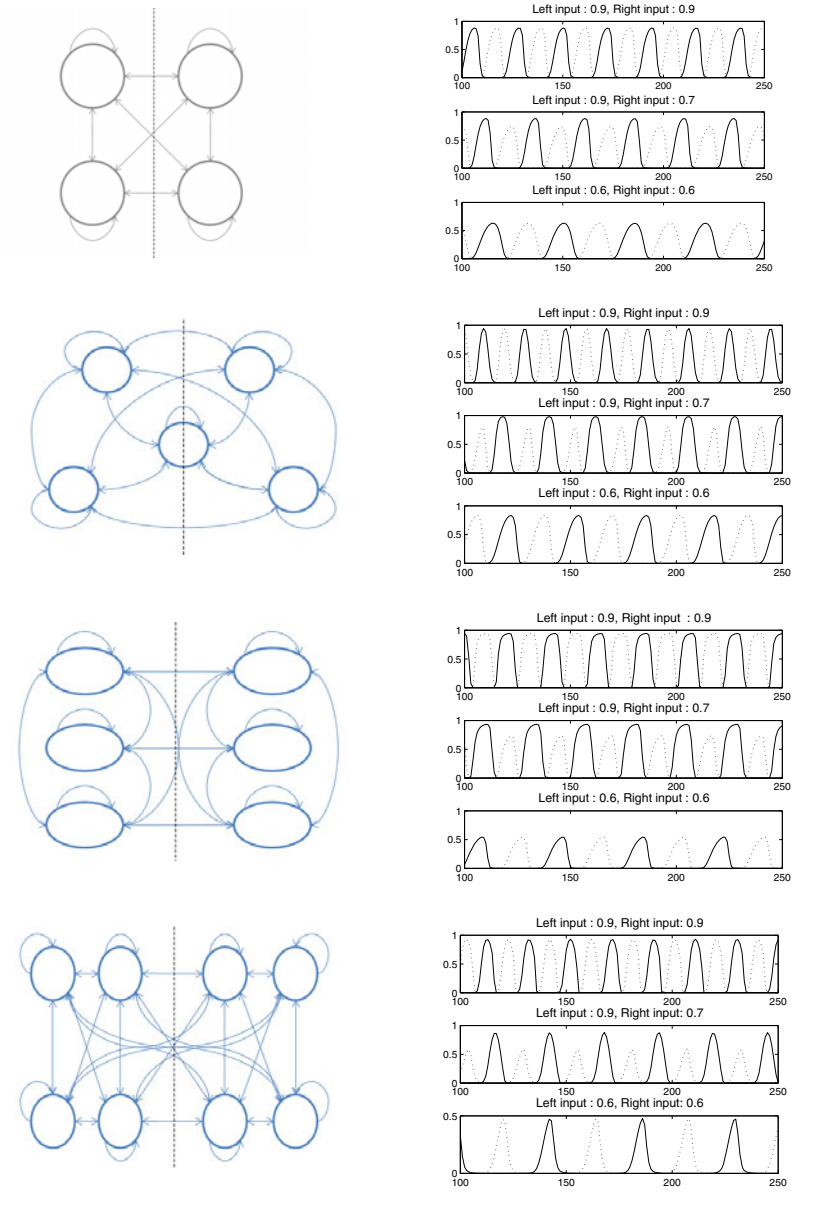


Fig. 3. Evolved neuron models with multiple neurons

for the two motor signals. The fitness level F_2 for two or three neurons is not larger than 0.5, which means its maximum phase difference is smaller than 90°. In contrast, four neurons or more produce the anti-phase signals. It means at least four neurons are required for the property. The input/output response also shows better performance with a larger number of neurons. The characteristics

of frequency variability is significantly influenced by the number of neurons. Four neurons or more are effective for this performance.

Fig. 2 shows frequency variability depending on the number of neurons,. Frequency variability range is not significantly different among a varying number of neurons, starting from four neurons. However, more neurons tend to generate a larger range of frequencies with the input change from 0.5 to 1. Here, the input value 0.5 or less does not generate the osciilations. The range of frequencies is estimated as the difference between the maximum and minimum frequency obtained when the whole range of inputs from 0 to 1 was tested. For example, four, five, six, eight neurons have maximal range of frequencies, 3.0 Hz, 3.6 Hz, 4.2 Hz, 5.1 Hz, respectively. Thus, it is presumed that more neurons have a capability of producing a variety of frequencies. When we give the same input signals to the symmetric neurons, frequency variability is improved. Our experiments with multiple neurons show similar results with Ijspeert et al.'s CPGs for lamprey animals [7], which have symmetric structure and frequency variability.

Fig. 3 shows the CPG model with multiple neurons. They produced high fitness values, that is, anti-phase property, large frequency variability, and good input/output response.

We additionally tested the neuron model with asymmetric structure (not shown here). Asymmetric structure increases the phase shift and frequency variability. The phase difference between the left and right motor actions was increased up to larger than 90 degrees, but the input/output response rate did not improve. The frequency range becomes larger for the asymmetric structure.

4 Discussion

We considered the phase difference 180 degrees between left and right motor actions for desired CPG neuron model. The anti-phase output is not necessarily required for every CPG model, but mostly controlling a multi-legged robot requires anti-phase for the left and right motors. The anti-phase CPG model is used in quadruped robot [4], hexapod robot [5], lamprey [6,7] and a multi-legged robot with more than six legs [10]. Some CPG models may need more motor outputs than two motors. Evolving the CPGs with multi-outputs is more difficult, and we can obtain desired neuron circuits by combining several simple CPG modules.

We used four criteria in the fitness function to derive the CPG characteristics over multiple neurons. Our approach follows incremental evolution in the fitness evaluation. Possibly we can add more characteristics for the CPG evolution, for example, more variety of input/output control, more variety of phases, and more control/input variables for phase and frequencies.

5 Conclusion

In this paper we investigate the capability of a small number of neurons for the central pattern generator characteristics, using evolutionary computation.

Genetic algorithms evolved neural parameters for the CTRNN neural structure. We use four criteria, oscillation characteristics, phase difference between the left and right motor signals, the input/output response, and frequency variability. We infer from the experiments that four neurons can develop those CPG characteristics while two or three neurons have their limitation of phase difference and frequency variability. Also more neurons have a tendency of improving the frequency variability by input signals.

Acknowledgments. This research has been by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Centre) support program supervised by the IITA (Institute for Information Technology Advancement) (IITA-2008-C1090-0803-0006).

References

1. Delcomyn, F.: Neural basis of rhythmic behavior in animals. Science 210(4469), 492
2. Lewis, M., Etienne-Cummings, R., Hartmann, M., Xu, Z., Cohen, A.: An in silico central pattern generator: silicon oscillator, coupling, entrainment, and physical computation. Biological Cybernetics 88(2), 137–151 (2003)
3. Taga, G.: A model of the neuro-musculo-skeletal system for human locomotion. Biological Cybernetics 73(2), 97–111 (1995)
4. Billard, A., Ijspeert, A.: Biologically inspired neural controllers for motor control in aquadruped robot. In: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, vol. 6 (2000)
5. Lewis, M., Fagg, A., Bekey, G.: Genetic algorithms for gait synthesis in a hexapod robot. Recent Trends in Mobile Robots, 317–331 (1994)
6. Ekeberg, Ö.: A combined neuronal and mechanical model of fish swimming. Biological cybernetics 69(5), 363–374 (1993)
7. Ijspeert, A., Hallam, J., Willshaw, D.: Artificial lampreys: Comparing naturally and artificially evolved swimming controllers. In: Proceedings of the Fourth European Conference on Artificial Life, pp. 256–265. MIT Press, Cambridge (1997)
8. Ijspeert, A., Crespi, A., Ryczko, D., Cabelguen, J.: From swimming to walking with a salamander robot driven by a spinal cord model. Science 315(5817), 1416 (2007)
9. Beer, R., Gallagher, J.: Evolving dynamic neural networks for adaptive behavior. Adaptive Behavior 1(1), 91–122
10. Tsujita, K., Tsuchiya, K., Onat, A., Aoi, S., Kawakami, M.: Locomotion control of a multipod locomotion robot with CPG principles. In: Proc. of The Sixth International Symposium of Artifical Life and Robotics, vol. 2, pp. 421–426 (2001)

Toward Minimally Social Behavior: Social Psychology Meets Evolutionary Robotics

Tom Froese and Ezequiel A. Di Paolo

CCNR, University of Sussex, Brighton, UK
t.froese@gmail.com, ezequiel@sussex.ac.uk

Abstract. We report on a set of minimalist modeling experiments that extends previous work on the dynamics of social interaction. We used an evolutionary robotics approach to fine-tune the design of a recent psychological experiment, as well as to synthesize a solution that gives clues about how humans might perform under these novel conditions. In this manner we were able to generate a number of hypotheses that are open to verification by future experiments in social psychology. In particular, the results indicate some of the advantages and disadvantages of relying on social factors for solving behavioral tasks.

Keywords: Evolutionary robotics, social psychology, social interaction.

1 Introduction

Since its beginnings in the early 1990s evolutionary robotics (ER) has established itself as a viable methodology for synthesizing models of 'minimally cognitive behavior', namely the simplest behavior that raises issues of genuine cognitive interest [2]. Within this context there has been a growing interest in using this method to investigate the minimal dynamics of social interaction (cf. [5] for a review). As a specialization of the ER methodology, we can conceptualize this kind of modeling as investigations into the dynamics of 'minimally *social* behavior'.

What is interesting about some of these recent advances in ER is that the synthetic method has been used to create models which are explicitly inspired by actual psychological experiments. Moreover, some of these models have been specifically designed to generate insights with the potential to generate mutually informing collaborations between the field of artificial life and the traditional empirical sciences, especially social psychology (e.g. [3, 4]). One promising target for this endeavor is Auvray, et al.'s [1] minimalist perceptual crossing experiment. This psychological study attempts to explore the most basic conditions necessary for participants to recognize each other through minimal technologically mediated interaction in a shared virtual space. Since this study will be the target of the modeling experiments presented in this paper we will describe the study in a bit more detail here. A schematic of the overall experimental setup is shown in Figure 1.

Two adult participants, acting under the same conditions, can move a cursor left and right along a shared 1-D virtual tape that wraps around. They are asked to indicate the presence of the other partner. The participants are blindfolded and all they

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 426–433, 2011.

can sense are on/off tactile stimulations on a finger when their cursor crosses an object on the tape. Apart from each other, participants can encounter a static object on the tape, or a displaced 'shadow image' of the partner, which is strictly identical to the partner as regards to size and movement characteristics. There are thus three distinct types of objects (each is 4 pixels wide) which can be encountered by a participant, one of which is placed at a fixed location and two of which are moving within the 1-D space. The two mobile objects exhibit exactly the same movement, but only an overlap of the receptor fields of both participants gives rise to mutual sensory stimulation. Note that the difference between these three types of objects cannot be directly provided by the sensors, which in all cases can only produce a binary response depending on whether something is overlapping the receptor field or not.

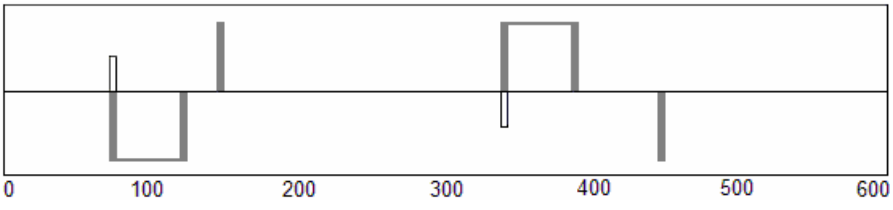


Fig. 1. The experimental setup: Two participants capable of horizontal movement face each other in a 600 unit 1-D 'tape' that wraps around at the edges. Note that a participant's receptor field (white bar) can encounter three different objects (gray bars): a static object (located at 148 or 448, depending on the participant), the other participant's avatar (coinciding with the location of its receptor field), and the other's 'shadow' (attached to the other's avatar via a rigid 48 unit link). Since all objects have the same width (4 units) they cannot be told apart by any difference between their appearances (they all give rise to an all-or-nothing tactile response).

The results of the psychological study show that, at least under the minimalist conditions of this experiment, the successful recognition of an ongoing interaction with another person is not only based on individual capacities. It is also based on certain properties that are intrinsic to the joint perceptual activity itself. The important issue is that the scanning of an object encountered will only stabilize in the case that both partners are in contact with each other —if interaction is only one-way, between a participant and the other's shadow, the shadow will eventually move away, because the participant it is shadowing is still engaged in searching activity. Two-way mutual scanning is the only globally stable condition. Therefore, the solution to the task does not only rely on individuals performing the right kind of perceptual discrimination between different sensory patterns (the empirical data shows they cannot distinguish between the other and its shadow), but also emerges from the mutual perceptual activity of the experimental subjects that is oriented towards each other.

2 Previous Work

Di Paolo, et al. [3] used ER to generate a simulation model of the perceptual crossing experiment, and successfully replicated the results while at the same time gaining some additional insights into the dynamics of the interaction process. For example,

the problems that their agents had with avoiding interactions with their respective static objects led them to predict similar difficulties for human participants. It turns out that repeated crossing of an object produces a pattern of stimulations located on the same spatial position. This is the same in the case of a fixed object and the other agent if it moves in coordinated anti-phase. Accordingly, it is difficult for the two patterns to be distinguished (see Fig. 3 for an example). This prediction was supported by the empirical data presented by Auvray and colleagues [1], but previously went unnoticed. Froese and Di Paolo [5] replicated these modeling results and introduced some variations, which resulted in further hypotheses about the stability of the interaction process in organizing the behavior of the interactors.

In both modeling studies [3, 5] it was demonstrated that the simulated agents make use of the *duration of contact* with objects in order to discriminate interactions with a static object (always same length of stimulation for same velocity) and the other agent (potentially shorter or longer stimulation, depending on whether the other passes by in in-phase or anti-phase movement). This is a reliable basis of distinction because the other agent is always moving. It is unlikely, however, that this is the main strategy employed by humans in the original psychological study. To be sure, during the training phase the participants were asked to interact with a four-pixel wide object in three conditions. The target object was either (i) static, (ii) moving at a constant speed of 15 units/second, or (iii) moving at a constant speed of 30 units/second, and each of these one min. training phases was announced as such. It could thus be possible that the participants learned the correlation between contact duration and whether an object is static or moving. In practice, however, the difference in duration is small enough such that it is unlikely to be the main strategy of the participants, though there is some evidence that contact duration made a difference, leading to 31.3% of clicking response (cf. event E6 [1, p. 40]). Still, we hypothesize that the successful behavior of the participants is based on different types of interactions afforded by the static object and the other active participant, rather than their differing durations of contact.

The question we want to address in this paper is: can we use ER to investigate the kinds of strategies that are available when such a duration-based strategy is excluded from the experimental design? One way to approach this is to make all objects (i.e. agents, shadow objects, and static objects) within the virtual environment infinitely small. This can be done by simply checking whether the sign of the difference of the locations (of the agent and some target object) has changed compared to the previous time step. If the sign has changed, then we activate the agent's receptor field. Since in this case all objects afford an equal duration of contact (i.e. 1 time step), it is no longer possible for the agents to trivially rely on the fact that other moving objects entail a shorter contact. Can we use ER to generate a strategy that enables the agents to successfully locate each other even under this more ambiguous situation?

3 Further Experiments in Perceptual Crossing

The simulation model includes two agents facing each other in a 1-D environment (i.e. one agent faces 'up' and one agent faces 'down'), which wraps around on itself after 600 units of space. In the simulation all distance and time units are of an arbitrary scale. There is no noise. Each model agent controls the horizontal movement of

its 'body', i.e. the position of its receptor field. The position of the agents is represented by continuous variables. The velocity of each agent is determined by taking the difference in output of two nodes of a continuous-time recurrent neural network (CTRNN), as described by Beer [2]. The CTRNN is fully inter-connected with self-connections. No symmetry is imposed on the network. The sensory input of an agent is activated (set to 1) when its receptor field passes another object (i.e. it crosses it between two time steps), otherwise the input remains off (set to 0). Each node receives sensory input which is multiplied by a specific input gain.

The GA evolved 100 solutions spread throughout 10 niches for several thousand generations. Each solution was evaluated for 100 trials lasting for 800 units of time, each with a time step of 0.1; the overall score was weighted toward the worse trials. Each solution coded for a clonal pair of CTRNNs (range of biases and weights [-8, 8], time constants [1, 200]). For more details of the GA, see [5]. In contrast to the original experimental setup, each object in the environment, no matter whether it is moving or stationary, only activates the receptor field of a passing agent for 1 time step. To make the solutions more evolvable it was necessary to include a large range of input gains (range [-1000, 1000]) so as to compensate for the minimal period of stimulation; including a sensory delay of 5 units of time was also helpful. As in previous modeling experiments, the solutions were evaluated in terms of how close the two agents were to each other on average during a trial.

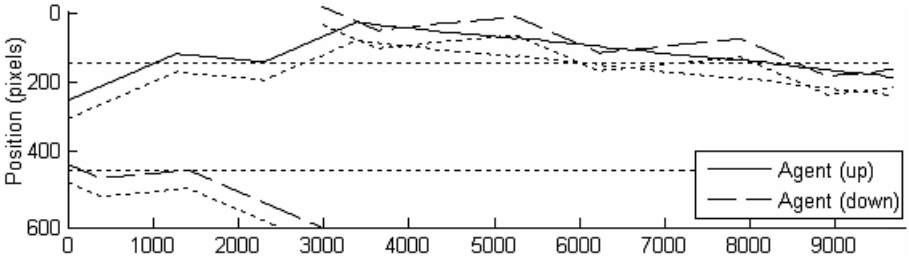


Fig. 2. A trial run showing the movement of the agents during the first 1000 time steps. The agents manage to avoid their respective static objects (after only 1 stimulation), eventually locate each other, and then continue to engage in perceptual crossing until the end of the trial.

Experiment 1: Making objects infinitely small.

We successfully evolved a 6-node CTRNNs to cope with this modified setup. To test the robustness of this solution we ran a comprehensive set of test trials; the average score is not significantly different from that of the original modeling setup. It turns out that the strategy of the agents is based on the close proximity of the two shadow objects. All the agents have to do is distinguish between one stimulation and two consecutive stimulations. This is a robust individual-based strategy to locate the other since: (i) passing the static object only causes *one* activation of the receptor, and (ii) passing the other agent with its attached shadow results in *two* activations. In other words, the evolution has found an individual-based solution that relies on an external factor, namely the relationship between the agents and their shadows (cf. Fig. 2).

Ironically, this behavioral strategy is robust because the shadow, which was meant to introduce an essential ambiguity into the experiment, has been appropriated to disambiguate the target from the static object. Is this a strategy that would be used by the human participants of the original study? Participants were indeed told about the experimental setup, including the three types of objects that they could encounter, but "the precise relation of the mobile lure yoked to the avatar was not explained" ([1], p. 38). Nevertheless, a large percentage of responses was preceded by a double stimulation (event E2, 32.3%, [1], p. 40), indicating that the shadow might have played a role in the positive empirical results. But what kind of strategies would be available if participants cannot take advantage of the agent-shadow relationship?

Experiment 2: Making shadows maximally distant.

We do not want to completely sever the link between the movements of the agents and their shadow objects, since this is an essential aspect of the experiment. Instead, we simply make the link between them maximally distant (150 units)[1]. The rest of the setup remains the same as in the previous experiment. We evolved 6-node CTRNNs for several thousand generations and then chose the fittest solutions to run some test trials. The outcome of a typical trial is shown in Figure 3.

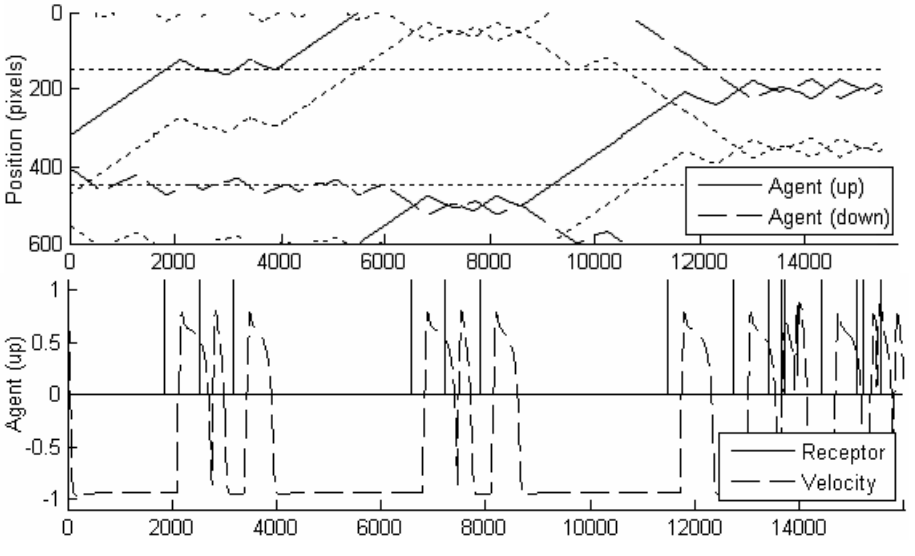


Fig. 3. Initial movement of the agents during a trial with maximally distant shadows. Note that it in this case it was impossible for agent 'up' to distinguish between the sensory-motor pattern produced by interacting with the static object (between ca. 2000-3000 time steps), and by interacting with the other agent (between ca. 7000-8000 time steps). Both encounters elicit the same three stimulations and the same motor response (bottom graph).

[1] Strictly speaking, in a 600 unit-wide circular world, being apart 300 units would be maximally distant. However, in this case there is another stable perceptual crossing situation, in which the agents can interact at a distance by stimulating each other with their shadow objects.

First, the agents explore their static objects for some time, then proceed to explore the rest of the space, then encounter each other and engage in some initial perceptual crossing. This mutual interaction breaks down after a while, and they continue exploring until they re-establish perceptual crossing at another location. Such break-downs occur more often than in the original setup, because agents are more likely to miss each other with infinitely small object sizes. Indeed, the possibility of coordination break-down could be a first indication that the agents have to be much more active and responsive in their interaction in order to disambiguate the situation. They cannot make use of persistent and reliable external factors to assess the viability of their behavior, and thus they are more open to commit errors and mistaken responses. The behavior of the agents during the trial run thus looks much more lifelike than that of previous solutions. This modeling experiment leads us to the prediction that the performance of human participants under these modified conditions would not be significantly different than from the original setup.

However, there still remains a problem in terms of this model. When the agents meet without receiving different stimulation beforehand, they engage on the basis of identical controllers (same structure and same internal state) such that they will mirror their behavior perfectly. This produces the same sensory-motor correlation as if they were oscillating around their static object. And since agents are more likely to encounter each other, evolution produces solutions which treat the occurrence of this sensory-motor pattern as always being due to the other rather than to the static object (a good choice, given the circumstances). However, occasionally this will result in both agents getting stuck on their static objects for the whole of a trial, giving rise to what looks like truly pathological behavior. It appears that such interactions do not break down when agents become entrained in too close proximity, thereby always making another contact with the object on their return path.

Experiment 3: Coordinated behavior.

How can we use ER to generate solutions that are better at distinguishing the other agent from the static object? As a first step, we remove the possibility of functionally identical CTRNNs encountering each other by simply activating the receptor field of a randomly chosen agent at the start of the trial, thus ensuring a minimal difference in individual histories. Moreover, in [4] we showed that sensitivity to social contingency can emerge from the interaction process if agents are required to coordinate their behaviors in a way that forces them to break the symmetry of their interactions. We therefore introduce an additional requirement into the fitness function by rewarding solutions in which agents travel together while continuing to engage in perceptual crossing. Since the agents are clones, this coordinated activity will require them to break the symmetry of their interactions in order to succeed. Moreover, since it is impossible to coordinate with the static object, we have emphasized the possibility of distinguishing the other in terms of its responsiveness.

We evolved agents with this modified fitness function that are able to coordinate their behavior so as to travel together while interacting (cf. Fig. 4). While engaging in perceptual crossing, the agents eventually start to drift together horizontally. In other words, even though the agents are structurally identical, have minimally different histories (internal states), and are not affected by noise during the trial, they are

nevertheless able to regulate the interaction such that the symmetry of their individual behaviors is broken. In fact, when one of the agents encounters its static object during this coordination process, the agents are able to re-negotiate the direction of drift and return the other way, much like what was found in the pioneering work by Quinn, et al. (2003). Future work could analyze in more detail the precise manner in which this symmetry breaking is realized in the dynamics of the interaction process.

Nevertheless, it is still the case that this strategy is not as robust as the solutions which we have excluded from the experimental design. If both agents happen to encounter their static objects at the start of the trial, they are likely to continue oscillating around it until the trial is terminated, even if the possibility of functional identity has been removed in principle. At first sight this appears to be evidence that the agents are not sensitive to the social contingency of their interaction; otherwise they would presumably notice the lack of responsiveness of the static object and eventually move away. But this way of looking at the problem essentially demands an *individual* response alone, i.e. detecting lack of social contingency when there is none to be detected. In contrast, perhaps the existence of this pathological behavior is an indication of the truly *social* nature of the evolved solution? Indeed, if only one agent becomes trapped it will eventually be freed by the other agent, which entrains it in an interaction process such that they move away together.

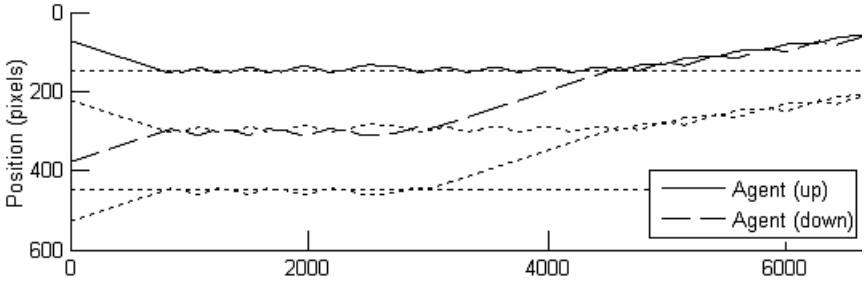


Fig. 4. Initial movements during a trial run showing coordinated behavior. First, both agents get stuck: agent 'up' on its static object and agent 'down' on the other's shadow. Then agent 'down' manages to break free and continues searching until the agents meet. This interaction frees agent 'up' from its entrainment and they start moving together until the end of the trial.

4 Discussion

We used ER as a process of fine-tuning the simulated setup to avoid the evolution of trivial solutions. At the same time it enabled the formulation of novel empirical predictions, in particular that those elements of the original experimental setup which were used as the basis for trivial solutions are not essential to the general results of that original study. Thus, we can hypothesize that the overall outcome of the original study will not be significantly altered when making the objects infinitely small and displacing the shadow object by 150 units. Indeed, we can venture a further hypothesis that part of the reason why the original study found less response to the static object, when compared to the two mobile objects, was that entrainment with this object was often broken by the actions of the other participant.

Another thing that we can learn from the design process is just how difficult it is to evolve a behavioral strategy that is only based social interaction. One important factor is that basing a behavioral strategy on the responsiveness of the other introduces an inherent risk factor in to the situation. What happens when the presumed 'other' does not react to your behavior in a suitable manner but your individual discriminatory ability depends on a certain kind of interaction? This appears to be the case in the 'pathological' behavior of the agents. More generally, the other's behavior can be influenced by your own actions, but it evades your direct control in principle. Another factor is that detecting another's responsiveness *as such* during an interaction is a much more demanding task than detecting simple environmental cues (e.g. difference in stimulus duration, difference in number of contacts, difference in noise, etc.).

Finally, the modeling experiments presented here point to the possibility of future dialogue between ER and cognitive science. The fact that the agents in the final experiment are unable to distinguish between the static object and the other agent individually, but can do so when that other agent is present, deserves further study, especially in relation to empirical findings. For example, studies of rehabilitation after brain damage have shown that patients often (i) find it impossible to individually achieve sensory-motor tasks in an abstract context, (ii) have difficulty with them in a pragmatic context, and (iii) can function normally in socially situated circumstances (cf. Gallagher & Marcel 1999). A modeling hypothesis we can draw from these empirical findings is that discrimination of the static object will be possible for individual agents with more complex CTRNN controllers. Indeed, some preliminary experiments with 10-node CTRNNs have indicated this to be the case.

References

1. Auvray, M., Lenay, C., Stewart, J.: Perceptual interactions in a minimalist virtual environment. New Ideas in Psychology 27(1), 32–47 (2009)
2. Beer, R.D.: Toward the evolution of dynamical neural networks for minimally cognitive behavior. In: Maes, P., et al. (eds.) Proc. of the 4th Int. Conf. on Simulation of Adaptive Behavior, pp. 421–429. The MIT Press, Cambridge (1996)
3. Di Paolo, E.A., Rohde, M., Iizuka, H.: Sensitivity to social contingency or stability of interaction? Modelling the dynamics of perceptual crossing. New Ideas in Psychology 26(2), 278–294 (2008)
4. Froese, T., Di Paolo, E.A.: Stability of coordination requires mutuality of interaction in a model of embodied agents. In: Asada, M., Hallam, J.C.T., Meyer, J.-A., Tani, J. (eds.) SAB 2008. LNCS (LNAI), vol. 5040, pp. 52–61. Springer, Heidelberg (2008)
5. Froese, T., Di Paolo, E.A.: Modeling social interaction as perceptual crossing: An investigation into the dynamics of the interaction process. Connection Science (in press)
6. Gallagher, S., Marcel, A.J.: The self in contextualized action. Journal of Consciousness Studies 6(6), 4–30 (1999)
7. Quinn, M., Smith, L., Mayley, G., Husbands, P.: Evolving controllers for a homogeneous system of physical robots: structured cooperation with minimal sensors. Phil. Trans. R. Soc. Lond. A 361, 2321–2343 (2003)

Emergence of Cooperation in Adaptive Social Networks with Behavioral Diversity

Sven Van Segbroeck[1,3], Francisco C. Santos[2,3], Tom Lenaerts[3], and Jorge M. Pacheco[4]

[1] COMO, Vrije Universiteit Brussel, Brussels, Belgium
[2] CoDE-IRIDIA, Université Libre de Bruxelles, Brussels, Belgium
[3] MLG, Université Libre de Bruxelles, Brussels, Belgium
[4] ATP-Group, CFTC & DF, University of Lisbon, Lisbon, Portugal

Abstract. Whether by nature or nurture, humans often respond differently when facing the same situation. Yet, the role of behavioral differences between individuals when immersed in their social network remains largely ignored in most problems of natural and social sciences. Here, we investigate how diversity in the way individuals assess their adverse social partners affects the evolution of cooperation. We resort to evolutionary game theory (EGT) to describe the dynamics of populations in which individuals interact according to an adaptive social network and may respond differently to unwanted social interactions. We show that increasing the number of ways of responding to adverse ties in the population always promotes cooperation. As such, adaptive social dynamics and behavioral differences benefit the entire community even though myopic individuals still act in their own interest. As defectors are wiped out, surviving cooperators maintain the full diversity of behavioral types, providing the means to establish cooperation as a robust evolutionary strategy.

Keywords: Evolutionary game theory, cooperation, collective behavior, complex networks.

1 Introduction

Cooperative behavior constitutes the hallmark of human society [8]. We tend to help others, even if providing such help is costly. However, often it is advantageous to accept all help offered by others without ever giving anything in return, creating the famous paradox of cooperation. The framework of EGT [7] conveniently formulates the problem by representing interactions between individuals in terms of simple games, like the two-person prisoner's dilemma [13], where each individual has the choice to either cooperate or defect. Individuals receive a certain payoff upon interaction, whose value depends on their own action and on that of their partner. The payoff they accumulate after interacting with all their contacts measures their fitness and represents their social success. Those that do well will be imitated and their behavior spreads in the population.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 434–441, 2011.

Often one considers a black and white world with only unconditional cooperators (*C's*) and unconditional defectors (*D's*). The fate of each strategy is determined by the payoff values characterizing the game being played. Mutual cooperation leads to a reward R for both individuals, mutual defection to a punishment P. A C interacting with D receives a suckers payoff S, whereas the D gets a temptation to defect T. The fingerprint of the prisoner's dilemma is associated with the payoff ranking $T > R > P > S$: joint cooperation is threatened by the temptation to defect towards a cooperator and by the fear of being betrayed by cooperating against a defector. In infinite, well-mixed populations evolution will always drive *C's* to extinction [4,2].

This scenario drastically changes when one takes the specific structure of modern networks of interaction and cooperation into account [10,16]. It has, for instance, been shown that heterogeneity in the role and position of individuals in social networks promotes strong levels of cooperation [14,16,17]. This effect even enhances when one recognizes that interaction networks are dynamic entities [6], whose structure co-evolves with the individual behavior [15,12,18]. Up to now, however, the *C's* and *D's* that continuously reshape their interaction network exhibit no differences in the way they manage their contacts. This situation contrasts with our everyday experience, where we recognize a continuous behavioral spectrum: Two *C's* (*D's*) may react differently when confronted with the same unfavorable situation. Their context in the social web will usually reflect a singular melting pot of influences, from culture, environment, family, acquaintances and friends, which together trigger a wide range of responses among individuals in the population [1]. In this work, we study how variability in the spectrum of possible reactions to adverse ties influences cooperation among humans along the links of the social web.

2 The Model

Consider a population described in terms of a network with constant number of nodes N. Every node represents an individual, every edge an interaction between the two individuals it connects. The number of edges changes in time as individuals continuously seek new interactions while abandoning old ones. The lifetime of each interaction depends on the behavior of both individuals involved, coupling the network dynamics with the behavioral (strategy) dynamics. Irrespective of one's *game strategy* (C or D), interacting with a C always leads to a higher payoff than interacting with a D ($S < R$ and $P < T$). Individuals will therefore be satisfied about their connections with *C's* and would like to maintain these as long as possible. Connections with *D's*, on the other hand, can be considered as adverse and will be broken at different rates, determined by the individuals' *linking strategy*. We introduce behavioral diversity by considering M (usually many) different linking strategies, and study the entangled co-evolution of game strategy and linking strategy with the self-organization of the population structure.

Let us denote the combined game and linking strategies as S_i, with $i \in \{1, \ldots, 2M\}$. We assume for simplicity that all individuals seek new interactions

with same propensity α, independent of their game or linking strategy. New links are therefore formed at rate α^2. Links connecting S_i individuals with S_j individuals disappear at rate $\kappa_{ij} = \frac{1}{2}(\gamma_{ij} + \gamma_{ji})$, where γ_{ij}^{-1} (γ_{ji}^{-1}) is the average time S_i (S_j) individuals would attribute to links with S_j (S_i) individuals. The value γ_{ij} reflects whether S_i individuals are satisfied or dissatisfied with their links with S_j individuals. When satisfied, γ_{ij} is taken as the minimum value γ among all γ_{ij}. When dissatisfied, on the other hand, γ_{ij} is given by the linking strategy of S_i individuals and satisfies $\gamma_{ij} \geq \gamma$. The corresponding linking dynamics of the network can be described by a set of ordinary differential equations [11,12]:

$$\dot{L}_{ij} = \alpha^2(N_{ij} - L_{ij}) - \kappa_{ij} L_{ij}, \tag{1}$$

where L_{ij} (respectively, N_{ij}) is the number (respectively, maximum number) of edges connecting S_i individuals with S_j individuals. These differential equations lead to a stationary distributions of links given by $L_{ij}^* = N_{ij}\phi_{ij}$, where $\phi_{ij} = \alpha^2(\alpha^2 + \kappa_{ij})^{-1}$ denotes the fraction of active links between S_i and S_j individuals.

Strategies spread in the population according to a mutation-selection process defined by the pairwise comparison rule [20,22]. At every strategy update, an individual A is drawn randomly from the population. With probability μ, he adopts a strategy selected randomly from all $2M$ available strategies. With probability $1 - \mu$, fitness determines whether he adopts the strategy of a randomly selected individual B, i.e., A imitates B with probability $p_{AB} = (1 + e^{\beta(f_A - f_B)})^{-1}$, where f_X denotes the payoff accumulated by player X after playing a one-shot game round with each neighbor in his network of contacts. The value β (≥ 0) controls the intensity of selection. The limit $\beta \to \infty$ leads to imitation dynamics in which the individual with the lower payoff always adopts the strategy of the one with the higher payoff. When $\beta \ll 1$, the process becomes equivalent to the weak-selection limit of the frequency dependent Moran process [9].

When the time scale for network updating (τ_a) is much faster than that for strategy updating (τ_e), the steady state of the linking dynamics determines the configuration of an individual's neighborhood every time he updates his strategy. In this limit, the fitness of an S_i individual is given by [12]:

$$f_i = \sum_j a_{ij}\phi_{ij}(N_j - \delta_{ij}), \tag{2}$$

where $A = [a_{ij}]_{i,j=1,\ldots,2M}$ is the game payoff matrix and N_j the number of S_j individuals. This is mathematically equivalent to the fitness of an S_i individual playing a game specified by the rescaled payoff matrix

$$B = [b_{ij}]_{i,j=1,\ldots,2M} = [a_{ij}\phi_{ij}]_{i,j=1,\ldots,2M} \tag{3}$$

in a finite, well-mixed population (complete network). This allows us to compute analytically the probability (fixation probability) ρ_{ij} that an individual with strategy S_i takes over a population of $N - 1$ individuals with strategy S_j, assuming that meanwhile no additional strategies appear because of mutations. Following [22],

$$\rho_{ij} = \frac{Erf(\xi_1) - Erf(\xi_0)}{Erf(\xi_N) - Erf(\xi_0)} \tag{4}$$

with $Erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x dy e^{-y^2}$, $\xi_k = \sqrt{\frac{\beta}{\mu}}(ku + v)$, $2u = b_{ii} - b_{ij} - b_{ji} + b_{jj}$ and $2v = b_{jj} + b_{ij}N - b_{jj}N - b_{ii}$. Computer simulations show that results obtained under the fast linking assumption ($\tau_a \ll \tau_e$) remain valid for a much wider range of time scales than one would expect from theoretical considerations [12].

As we address the role of behavioral diversity in this work, we will typically consider more than two possible strategies ($M > 1$), which precludes an analytical description of the stochastic evolutionary dynamics for arbitrary mutation rates. By assuming that mutations occur only rarely, we can analytically describe the system in a compact form [5,3]. The evolutionary dynamics does no longer proceed in the entire $2M$-dimensional strategy space, but only along its boundaries, where there are never more than two different strategies present simultaneously. Indeed, as long as no mutations occur, stochastic update dynamics always drives the population to a homogeneous state (monomorphic population), i.e., a state in which only individuals of one particular strategy survive. Assuming a monomorphic population, a specific new strategy will show up with probability $\frac{\mu}{2M-1}$ and to the extent that μ is sufficiently small, this mutant will go extinct or will fixate before any new mutation occurs. We can approximate this system by a Markov Chain with only $2M$ states, each state corresponding to a certain homogeneous state of the population. The probability that the appearance of a random mutant in a state with otherwise only S_i individuals moves the population to a state with only S_j individuals defines the transition probability between the corresponding states of the Markov Chain. These probabilities define the transition matrix Λ of the Markov Chain, which is given by $\Lambda = [\Lambda_{ij}]_{i,j=1,\ldots,2M}$, where $\Lambda_{ii} = 1 - \frac{1}{2M-1} \sum_{k=1,k\neq i}^{2M} \rho_{ki}$ and $\Lambda_{ij} = \frac{\rho_{ji}}{2M-1}$ $(j \neq i)$. The normalized left eigenvector of the unit eigenvalue of Λ defines the stationary distribution, i.e., the fraction of time the population spends in each of the available strategies. The stationary distributions obtained using this small-mutation approach also hold for larger mutation rates, as also shown in [3].

3 Results

The most simple configuration is the one in which there are only two different linking strategies ($M = 2$): Individuals break up adverse connections either at a slow rate γ_S, or at a fast rate γ_F. In combination with the game strategy we obtain a total of four different strategies: slow *C's* (*SC's*) and *D's* (*SD's*), whose adverse interactions last long, and fast *C's* (*FC's*) and *D's* (*FD's*), whose adverse interactions are short lived. Following Equation (3), we obtain the payoff matrix:

$$\begin{array}{c} \\ SC \\ FC \\ SD \\ FD \end{array} \begin{array}{c} \begin{array}{cccc} SC & FC & SD & FD \end{array} \\ \begin{pmatrix} R\phi_S & R\phi_S & S\phi_S & S\phi_S \\ R\phi_S & R\phi_S & S\phi_M & S\phi_M \\ T\phi_S & T\phi_M & P\phi_S & P\phi_M \\ T\phi_S & T\phi_M & P\phi_M & P\phi_F \end{pmatrix} \end{array}, \tag{5}$$

where $\phi_x = \alpha^2(\alpha^2 + \kappa_x)^{-1}$, with $\kappa_S = \gamma_S$, $\kappa_M = \frac{1}{2}(\gamma_S + \gamma_F)$ and $\kappa_F = \gamma_F$.

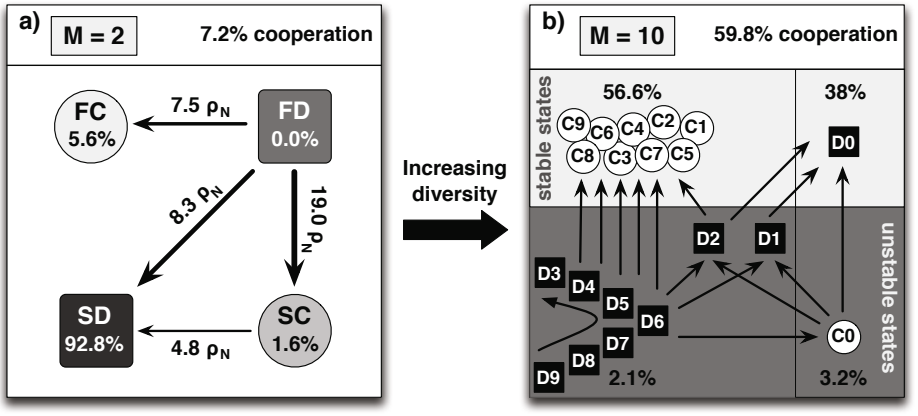


Fig. 1. Transition probabilities and stationary distributions for a population with M = 2 (a) and M = 10 (b) different linking strategies. In the limit of rare mutations, the dynamics reduces to transitions between homogeneous states of the population [5,3]. The arrows indicate those transitions for which the fixation probability is greater than neutral fixation, $\rho_N = \frac{1}{N}$. The explicit values were obtained analytically with the pairwise comparison rule [22]. **a)** Adaptive network dynamics allows *C's*, in the form of *SC* or *FC*, to remain in the population for 7.2% of the time. *D's* dominate because of the flow from *FD* to *SD*, either directly or by using the alternative route via *SC*. **b)** When M increases to 10, the population spends already 59.8% of the time in a cooperative state. Numbering *C's* and *D's* according to their type, *C0* (*D0*) being the slowest cooperator (defector), we see that increasing M splits the "outflow" of fast defectors among a wide range of different possibilities. As a result, cooperation emerges, since only few types of defectors are evolutionarily stable, whereas the vast majority of cooperative types work as "flow sinks". (N=100, β=0.01, T=2.1, R=2, S=0.9, P=1, α=0.4, δ=0.3)

Adopting the small-mutation approach discussed in the previous section, the complex co-evolutionary dynamics reduces to one associated with a Markov Chain with only 4 ($2M$) states. The Markov Chain's transitions that are favored by natural selection, i.e., those that are larger than $\rho_N = \frac{1}{N}$ (the fixation probability associated with neutral evolution), characterize the main driving forces of the evolutionary dynamics. These transitions are shown in Figure 1 for a region where *D's* dominate. The exact values are obtained by computing the probability that a mutant (with strategy located at the end of each arrow) fixates in a monomorphic population of individuals adopting the strategy located at the start of the arrow. We see that *SD's* are clearly the winners of the evolutionary race. *FD's*, on the other hand, are rendered disadvantageous with respect to any other strategy. When a *FC* manages to fixate, we end up in a rather stable scenario. In addition, *SC's* acquire a transient character, providing an alternative route from *FD* to *SD*. In this specific case, it is, however, mainly the direct transition from *FD's* into *SD's* that hinders *C's* survivability. As we will show below, the viability of *C's* relies on the extent to which the transition $FD \rightarrow SD$ is inhibited compared to transitions into *FC's*.

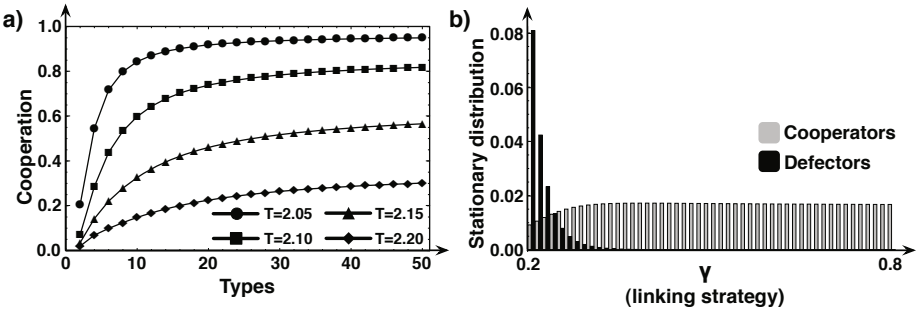


Fig. 2. Cooperation and behavioral diversity. **a)** The population spends more time in a cooperative state (of any type) when the number of possible types (M) increases, irrespective of the temptation to defect T in the prisoner's dilemma. (N=100, β=0.1, α=0.4, δ=0.3, R=2, P=1, S=3−T) **b)** While the majority of cooperator types is equally represented, only the slower types of defectors manage to survive. The defector population exhibits behavioral differences, which inhibits the dominance of slowest defector type, providing an escape hatch for cooperation to thrive (M=50, T=2.1, R=2, P=1, S=0.9).

We do now extend the analysis to an arbitrary number of linking strategies M, with rates for breaking adverse ties taken uniformly in the interval $[0.5-\delta, 0.5+\delta]$. This leads to a trivial generalization of payoff matrix (5). Figure 2a shows that cooperation blooms when the number of linking strategies M increases. In the following, we investigate the mechanism responsible for this remarkable performance of *C's*?

A first hint is provided in Figure 2b, where the fraction of time spent in each state of the population is shown for the case in which there are $M = 50$ possible types (linking strategies). Figure 2b shows that all types of *C's* are present in the population, whereas the *D's* who survive are only of the slowest types. Importantly, however, with increasing number of available types, the difference between the values of the rates associated with contiguous types is reduced, which provides a means for *D's* other than the slowest to survive in the population. These are precisely the *D's* who provide an escape hatch for *C's* to survive (cf. Figure 1a), since many of them will be disadvantageous with respect to *C's*. Indeed, Figure 2b shows that increasing the number of possible types, *D's* with break-up rates higher than the minimum γ are now able to survive. Although these individuals do not dominate, they effectively promote the appearance of fast *C's*, since the transition D→C between these types is favored by natural selection. On the other hand, because all C-types are neutral with respect to each other, they end up fairly equally distributed in the population, unlike *D's* for whom natural selection favors the slower types (cf. Figure 2b). Figure 1b also shows that, with increasing number of types, the number of C-types which are favored increases more than the corresponding number of D-types. As a result, all but the slowest *D's* are disadvantageous with respect to (most of the) *C's*. Increasing the number of types efficiently

inhibits the transition from the fastest to the slowest *D's*, paving the way for cooperation to thrive, an effect which remains valid irrespective of the model parameters, inasmuch as T and S are such that cooperators manage to survive. Mathematically, a large number of different types allows individuals to engage in a wider range of games and the complex set of different interactions allows the appearance of *D's* from which *C's* can profit.

4 Discussion

In social systems, individuals often behave differently when dealing with their social contacts. The present model shows that such diversity provides a perfect breeding ground for cooperation. Having individuals that think and behave differently creates a multi-dilemma environment [12,15] in which cooperation easily prevails, without any need for reputation, punishment or any other community enforcing mechanism.

Together with other recent results that underline the importance of different forms behavioral diversity in the evolution of cooperation [17,19,21], this work supports the idea that diversity deserves to be considered as a fundamental mechanism towards the emergence of cooperative behavior. In addition, given the strong multi-cultural nature and inherent diversity in modern societies, the current prevailing minority-friendly policies may have resulted from the evolutionary advantages shown here.

Finally, diversity in the way individuals organize their contacts may be important in other problems than the emergence of cooperative behavior. From spreading of infections diseases, in which individuals may react differently to a risk of infection from their neighbors, to spreading of computer viruses, where individual diversity in the resistance to pernicious attacks is common, the new mathematical framework presented in this article can be a valuable tool in the study of a broad spectrum of problems.

References

1. Buchan, N., Croson, R., Dawes, R.: Swift neighbors and persistent strangers: a cross-cultural investigation of trust and reciprocity in social exchange. Am. J. Soc. 108, 168 (2002)
2. Gintis, H.: Game Theory Evolving. Princeton University Press, Princeton (2000)
3. Hauert, C., Traulsen, A., Brandt, H., Nowak, M.A., Sigmund, K.: Via freedom to coercion: The emergence of costly punishment. Science 316(5833), 1905 (2007)
4. Hofbauer, J., Sigmund, K.: Evolutionary Games and Population Dynamics. Cambridge University Press, Cambridge (1998)
5. Imhof, L.A., Fudenberg, D., Nowak, M.A.: Evolutionary cycles of cooperation and defection. P. Natl. Acad. Sci. U.S.A. 102(31), 10797 (2005)
6. Kossinets, G., Watts, D.J.: Empirical analysis of an evolving social network. Science 311(5757), 88 (2006)
7. Maynard Smith, J.: Evolution and the Theory of Games. Cambridge University Press, Cambridge (1982)

8. Nowak, M.A.: Five rules for the evolution of cooperation. Science 314, 1560–1563 (2006)
9. Nowak, M.A., Sasaki, A., Taylor, C., Fudenberg, D.: Emergence of cooperation and evolutionary stability in finite populations. Nature 428, 646 (2004)
10. Ohtsuki, H., Hauert, C., Lieberman, E., Nowak, M.A.: A simple rule for the evolution of cooperation on graphs. Nature 441, 502–505 (2006)
11. Pacheco, J.M., Traulsen, A., Nowak, M.A.: Active linking in evolutionary games. J. Theor. Biol. 243, 437 (2006)
12. Pacheco, J.M., Traulsen, A., Nowak, M.A.: Coevolution of strategy and structure in complex networks with dynamical linking. Phys. Rev. Lett. 97, 258103 (2006)
13. Rapoport, A., Chammah, A.M.: Prisoner's Dilemma. Univ. of Michigan Press, Ann Arbor (1965)
14. Santos, F.C., Pacheco, J.M.: Scale-free networks provide a unifying framework for the emergence of cooperation. Phys. Rev. Lett. 95, 98104 (2005)
15. Santos, F.C., Pacheco, J.M., Lenaerts, T.: Cooperation prevails when individuals adjust their social ties. PLoS Comput. Biol. 2(10), e140 (2006)
16. Santos, F.C., Pacheco, J.M., Lenaerts, T.: Evolutionary dynamics of social dilemmas in structured heterogeneous populations. P. Natl. Acad. Sci. U.S.A. 103(9), 3490 (2006)
17. Santos, F.C., Santos, M.D., Pacheco, J.M.: Social diversity promotes the emergence of cooperation in public goods games. Nature 454, 213 (2008)
18. Van Segbroeck, S., Santos, F.C., Nowe, A., Pacheco, J.M., Lenaerts, T.: The evolution of prompt reaction to adverse ties. BMC Evol. Biol. 8(287) (2008)
19. Szabó, G., Fáth, G.: Evolutionary games on graphs. Phys. Rep. 446(4-6), 97 (2007)
20. Szabó, G., Tőke, C.: Evolutionary Prisoner's Dilemma game on a square lattice. Phys. Rev. E 58, 69 (1998)
21. Szolnoki, A., Perc, M., Szabó, G.: Diversity of reproduction rate supports cooperation in the prisoner's dilemma game on complex networks. Eur. Phys. J. B 61, 505 (2008)
22. Traulsen, A., Nowak, M.A., Pacheco, J.M.: Stochastic dynamics of invasion and fixation. Phys. Rev. E 74, 11909 (2006)

Evolving for Creativity: Maximizing Complexity in a Self-organized Multi-particle System*

Heiko Hamann, Thomas Schmickl, and Karl Crailsheim

Artificial Life Lab of the Department of Zoology, Karl-Franzens University Graz, Universitätsplatz 2, A-8010 Graz, Austria,
{heiko.hamann,thomas.schmickl,karl.crailsheim}@uni-graz.at

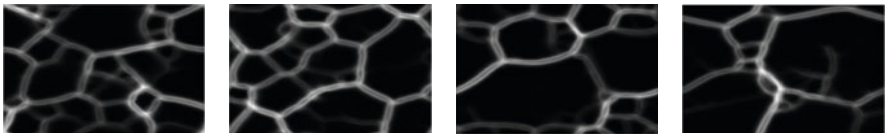
Abstract. We investigate an artificial self-organizing multi-particle (also multi-agent or swarm) system consisting of many (up to 10^3) reactive, mobile agents. The agents' movements are governed by a few simple rules and interact indirectly via a pheromone field. The system generates a wide variety of complex patterns. For some parameter settings this system shows a notable property: seemingly never-ending, dynamic formation and reconfiguration of complex patterns. For other settings, however, the system degenerates and converges after a transient to patterns of low complexity. Therefore, we consider this model as an example of a class of self-organizing systems that show complex behavior mainly in the transient. In a first case study, we inspect the possibility of using a standard genetic algorithm to prolongate the transients. We present first promising results and investigate the evolved system.

1 Introduction

Self-organizing systems such as natural or artificial organisms and swarms often form complex patterns [1,2,3]. In nature, these systems are subject to natural selection, which evolves complex patterns by adapting simple behavioral rules followed by the agents. In our work, we model similar complex systems and adapt them by applying Evolutionary Computation. These systems are interpreted as nonlinear dynamic systems that converge to a fixed point or to periodic behavior. Some systems, however, show seemingly stable (i.e., fixed point or periodic behavior) complex patterns although relevant steady states do not exist. Such systems rely on quasi-stationary (or quasi-periodic) states that are induced by long transients. The state of a system is said to be *transient* in the time segment between the initialization of the system and before a steady state (e.g., a fixed point, a periodic behavior, or a chaotic attractor) is reached. Examples of such systems showing quasi-stationary behavior are models of ecological systems [4,5] and many phenomena surrounding us in our everyday life such as ourselves [6]. Typically, the transient shows a much more complex and interesting behavior compared to the steady state, the system is actually slowly converging to.

* Supported by: EU-IST-FET project 'SYMBRION', no. 216342; EU-ICT project 'REPLICATOR', no. 216240.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 442–449, 2011.



(a) $t = 1 \times 10^5$. (b) $t = 3 \times 10^5$. (c) $t = 5 \times 10^5$. (d) $t = 6 \times 10^5$.

Fig. 1. One pattern out of the variety of many of the investigated system ($N = 3000$, $L = 300$, $\eta = 0.04$, parameters described below)

In this paper, we use a simple model of a multi-agent system. The agents move according to a few simple rules and interact indirectly via a (virtual) pheromone field. This model serves as an example of self-organized systems that depend on long transients. Based on only two parameters (resolution of the pheromone field discretization and/or the pheromone diffusion) we are able to change the behavior between relatively short transients (i.e., relatively fast convergence to a steady state of low complexity) and supposedly very long transients (actually so long that we even cannot be sure whether they are transient at all). We artificially restrict the system to rather low resolutions of the discretization leading to mostly manageable transients that are numerically simulated within a reasonable time. A first investigation of the transients was reported in [7]. Here, we try to prolongate the transients and to increase the complexity of the patterns by adjusting other parameters of the model. This is done by using a standard genetic algorithm [8]. Thus, we evolve parameter settings that result in long transients and high complexity which corresponds to increasing the time a self-organized system shows complex behavior.

We propose that Evolutionary Computation is a suitable tool to shape complex multi-particle systems in a desired manner. Together with the intrinsic creative potential of such particle systems (see Fig. 1), both techniques in combination serve as a powerful on-demand pattern generator. The objective of this work is, on the one hand, to produce a multi-particle system showing dynamic pattern formation, that might be applied in many ways, for example, in a novel robot controller concept or to generate pseudo-random textures for computer graphics (e.g., similar to Perlin's noise [9]). On the other hand, we want to determine the relevance of transients in self-organized systems and how to manipulate them.

2 Mathematical Model

In the following, we define the model of the investigated artificial multi-agent system as reported before in [7]. This system may also be called multi-particle system or swarm as it consists of many (up to 10^3) reactive agents (automata that map receptions to actions without an internal world model). The model used here is based on and is very similar to the model of complex transport networks reported by Jeff Jones in [10]. The main differences between the models are a

semi-continuous representation of agent data in Jones' model compared to a continuous representation in our model and a different density control (based on occupied patches in Jones' model, based on maximal pheromone field values here). Time is discrete in both models. Finally, both models are reduced to fully discrete models, because they are simulated on digital computers. However, the different methods of discretization do matter (rough discretization in patches and floating point arithmetic). The agents move in two-dimensional space with periodic boundary conditions (torus). The change of an agent's position $\mathbf{x}$ is defined by

$$\frac{d\mathbf{x}}{dt} = \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix} v, \tag{1}$$

for a velocity set to a constant value $v > 0$, except for the case that the local pheromone value is above a threshold P_{max} (then we set $v = 0$). The change of the agent's direction ϕ is defined by

$$\frac{d\phi}{dt} = \alpha(s_l(t), s_c(t), s_r(t))\gamma(t), \tag{2}$$

for $\alpha(s_l(t), s_c(t), s_r(t)) \in \{1, 0, -1\}$ defining the direction of the turns (clockwise, no turn, or counterclockwise), γ defining the absolute value of the turn angles, and for sensor values s_c and s_l that are defined by

$$s_c(t) = P\begin{pmatrix} x_1 + \cos(\phi)d \\ x_2 + \sin(\phi)d \end{pmatrix}, \tag{3}$$

$$s_l(t) = P\begin{pmatrix} x_1 + \cos(\phi - \psi)d \\ x_2 + \sin(\phi - \psi)d \end{pmatrix}, \tag{4}$$

for a pheromone field P representing the environment, $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, sensor angle ψ, and sensor distance d. s_r is defined analog to s_l. Closely following [10], we define

$$\alpha(s_l(t), s_c(t), s_r(t)) = \begin{cases} 0, & \text{for } s_c(t) > s_l(t) \\ & \wedge s_c(t) > s_r(t) \text{ (no turn)} \\ \pm 1, & \text{for } s_c(t) < s_l(t) \\ & \wedge s_c(t) < s_r(t) \text{ (random turn)} \\ +1, & \text{for } s_l(t) < s_r(t) \text{ (right turn)} \\ -1, & \text{for } s_r(t) < s_l(t) \text{ (left turn)} \end{cases}, \tag{5}$$

whereas the order (from top to bottom) of the conditions matters. The random turn has a probability of 50% for $+1$ and 50% for -1. We define

$$\gamma(t) = \begin{cases} \phi_{\text{rot}}, & \text{for } t \in \{0, \tau, 2\tau, \dots\} \\ 0, & \text{else} \end{cases}, \tag{6}$$

for a constant rotation angle ϕ_{rot} and a time interval τ at which the agents turn and their directions are updated (in this work we set $\tau = 1$). Thus, we obtain a synchronized system that is discrete in time. The system could, for example, be extended by defining γ as a stochastic process. This would transform Eq. 2 into a stochastic differential equation.

The pheromone field P is, in principle, defined by the standard diffusion equation

$$\frac{\partial P(\mathbf{x},t)}{\partial t} = D\nabla^2 P(\mathbf{x},t) - \eta P(\mathbf{x},t) + \theta \sum_{i=1}^{N} \delta(\mathbf{x} - \mathbf{x}_i(t)), \tag{7}$$

for diffusion D, evaporation rate η, addition θ (the Dirac delta indicates that an agent only contributes to the pheromone field at its position), number of agents N, and agent positions $\mathbf{x}_i(t)$. However, for simplicity and to reduce the computational complexity the diffusion and the evaporation are only executed at $t \in \{0, 10\tau, 20\tau, \dots\}$ unlike the addition process that is executed at $t \in \{0, \tau, 2\tau, \dots\}$.

As discussed above, the system is simulated on a digital computer. Thus, the pheromone field needs to be discretized. This is done by a grid. Here, the grid is chosen to be always quadratic and the number of grid points is given by L^2. In our implementation, the diffusion is not normalized in correspondence to the resolution of the grid (as it would be necessary in classical numerics).

If provided with a sufficient resolution of the grid, that is discretizing the pheromone field, the model shows a huge variety of complex patterns [7]. These patterns form, collapse partially, form again, and seem never to recur. This way the system shows high creativity that is very different compared to many other self-organizing systems that converge (quickly) to a steady state. This is related to the concept of synergetics where modes of high dynamics are governed by modes of slow dynamics [11]. Throughout this work, the agents' positions are initialized by a random uniform distribution throughout the whole space.

Table 1. Standard parameters

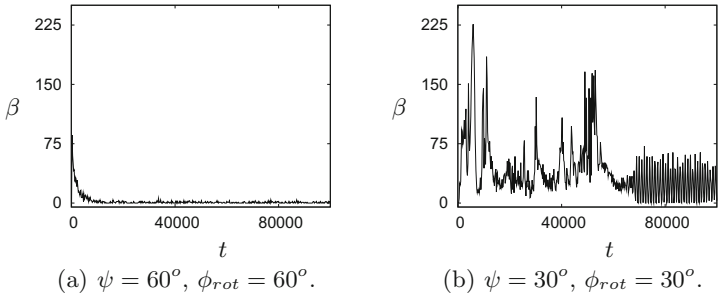
sensor angle ψ	45^o
rotation angle ϕ_{rot}	45^o
rotation period τ	1 [time units]
surface area of the torus s^2	1×1[length units]^2
grid length L	150
grid resolution	$L/s = 150$[1/length units]
sensor distance d	0.035 [length units]
velocity v	0.01 [length units/τ]
diffusion D	0.1 [$(1/L)/(10\tau)$]
evaporation η	0.04 [$1/(10\tau)$]
addition θ	5 [$1/\tau$]
active cell threshold δ_{active}	30
max. pheromone value P_{max}	300
simulated steps	5×10^4 [time units]

See Fig. 1 for some examples of patterns formed in the pheromone field and see Table 1 for the parameters used, if not stated explicitly.

3 Complexity and Transients

In order to investigate the model concerning transients we need a method determining when a steady state is reached. In a high-dimensional system this is, generally, difficult. In addition, we want also to evolve complex and dynamic patterns. Thus, we restrict ourselves to a simpler (also concerning computational complexity) but effective method. As we know from experience with the simulator, the steady states are all of low complexity (patterns formed out of straight lines with two, one, or no bifurcations or only clusters). Instead of checking for a steady state it would suffice to find a metric that measures the complexity of the dynamics.

In this work, we use the following metric: We put a second grid (called *counting grid*) of lower resolution (here: 30×30) over the pheromone grid and count the grid points of the pheromone grid that are above a threshold δ_{active} (called *active cells*). If there are more than 50% active cells within one cell of the counting grid we mark it as 'on' (otherwise 'off'). It is sufficient to calculate the current state of the counting grid only every 200 time steps because the motion of the agents and the pattern dynamics occur on different time scales. Finally, the value of our complexity metric β is determined by counting the differences between the current counting grid and the one obtained 200 time steps before (thus, $\beta \in \{0, \ldots, 30^2\}$). See Fig. 2 for examples of the evolution of the complexity measure β for different parameter settings. Fig. 2(a) shows an example of a pattern that degenerates quickly to a collection of stationary clusters. Fig. 2(b) shows a pattern that degenerates to a traveling pattern (causing oscillations in β as it moves over the counting grid) of three lines and two bifurcations after a transient of about 66,000 time steps.

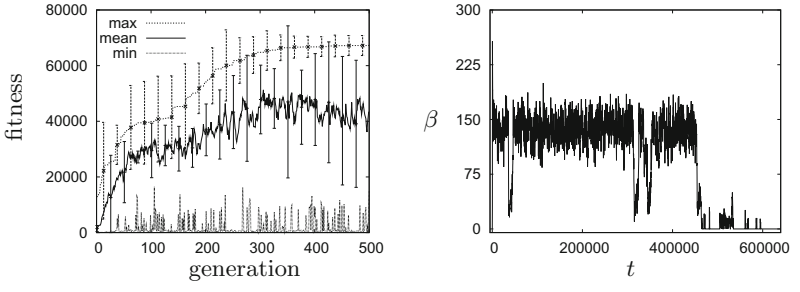


(a) $\psi = 60^o$, $\phi_{rot} = 60^o$. (b) $\psi = 30^o$, $\phi_{rot} = 30^o$.

Fig. 2. Examples of the temporal evolution of the complexity measure β ($v = 0.01$, $d = 0.06$, $\eta = 0.08$, $D = 0.05$, $N = 500$)

4 Evolving Complex and Long Transients

In this section, our new approach of evolving parameter sets that generate long transients with complex patterns is reported. The used optimizer is a standard genetic algorithm based on GALOPPS 3.2.4 by Erik Goodman [12]. Four parameters were varied within given intervals: rotation angle $\phi_{\text{rot}} \in [10^o, 90^o]$, sensor angle $\psi \in [10^o, 90^o]$, diffusion constant $D \in [10^{-3}, 10^{-1}]$, and number of agents $N \in [2^0, 2^{10}]$. The velocity of the particles was fixed because changes in the velocity are balanced by the diffusion constant as only the relation between these two parameters matters. The restriction to these intervals does only exclude parameter settings leading to irrelevant patterns (based on our experience). The values were encoded in a 10 bit Gray code. Mutations were single bit flips. Recombinations were two-point crossovers with leaving the 10-bit-groups intact. The selection mechanism was stochastic universal sampling. The population size was set to 20, the probability of a recombination was set to 0.05, and the probability of a mutation was set to 0.05. The fitness was defined as the sum of all changes in the counting grid until less than ten changes occurred for ten successive checks (i.e., 2,000 time steps). The simulation was stopped after 10^5 time steps in any case. For the used random uniform initialization of agent positions and directions this value was stochastic. The fitness was the mean of three simulation runs of the individual. A single run of 500 generations takes about three days of computing time on a contemporary desktop computer (single core). The averaged results of four runs of the genetic algorithm are shown in Fig. 3(a).



(a) Fitness over 500 generations. (b) Complexity measure over time.

Fig. 3. Left: max., mean, and min. fitness of four runs over 500 generations (errorbars indicate 95% confidence intervals). Right: Temporal evolution of the complexity measure β of the best individual found.

Beginning with a mean fitness of 1,446, a peak mean fitness of 51,293 was reached after 306 generations. The mean fitness of the best individual after 500 generations was 67,393. The best individuals of the four runs were very similar; we give the averages and the standard deviations: $\phi_{\text{rot}} = 89.4^o \pm 0.39$, $\psi = 10.1^o \pm 0.16$, $D = 0.00125 \pm 5 \times 10^{-4}$, and $N = 582.5 \pm 38.2$.

(a) Chaotic phase. (b) Expanding rect. (c) Colliding rect. (d) Chaotic phase.

Fig. 4. Pheromone field of best individual

The following analysis of the best individual shows the high degree of adaptivity to the fitness function. The evolution of the complexity measure for an example run is shown in Fig. 3(b). The dynamics in the counting grid were very high (cf. Fig. 2) with few interruptions. After about 450,000 time steps the pattern degenerates to a single line in this example (typically the observed transient was even longer). Thus, the pattern survives for the whole evaluation period (10^5 steps). The pheromone field of characteristic phases is plotted in Fig. 4.

The system shows, for most of the time, a rather chaotic behavior, see Fig. 4(a). However, about two times per 300,000 time steps it forms an expanding rectangle, see Fig. 4(b) to 4(d). The agents deform this rectangle because of the rotation angle $\phi_{\text{rot}} = 89.609^o \neq 90^o$. Through a non-trivial process this leads to an expansion which continues until the rectangle collides with itself (space is toroidal). Then another chaotic phases begins. These rectangle-formation-phases are clearly identified in Fig. 3(b) as the short phases of low values ($10 < \beta < 70$). The high fitness of this individual does not depend on the toroidal form of space as observed in runs of the genetic algorithm with strict bounds at all four sides (non-toroidal, data not shown).

5 Conclusions and Outlook

In this paper, we applied genetic algorithms successfully to increase the creativity and complexity of self-organizing systems by prolongating the transient. We investigated a multi-agent system with simple rules that shows a huge variety of complex patterns. This model is interpreted as an example of self-organized systems that rely on long transients because the investigated systems only show complex behavior before the steady state is reached. Parameter settings were evolved that lead to longer transients and, thus, maximize the time complex behavior is observed. The presented model is a microscopic (bottom-up) model inspired by slime molds [10]. The observed patterns bear similarities to macroscopic (top-down) differential equation models of slime molds as reported in [13]. Other relevant models showing similar patterns are the macroscopic reaction-diffusion models of animal coats [14,15]. In our microscopic model changes in the parameters result in a huge variety of patterns. Hence, small changes of parameters might result in big changes in the quality of the patterns. Thus, difficulties in evolving patterns with desired properties would be expected. In this paper, we have shaped the emergent properties of a self-organizing system

according to our requirements by evolution. The fitness landscape seemed to be good-natured and rather smooth because all four runs were successful and resulted in very similar individuals. In the future, we will further explore the abilities of multi-particle systems in combination with Evolutionary Computation. We will investigate those parameters in detail, which produce constantly changing, complex patterns. Our goal is to shape these patterns in a way that we are either able to manually construct them or to evolve them in a desired way (e.g., to connect predefined points in the modeled space or to obtain inhomogeneous textures). Possible applications for such dynamic pattern generators are ranging from artificial life, swarm robotics, and collective intelligence to arts and other forms of visual design and computer graphics.

References

1. Thompson, D.W.: On Growth and Form: The Complete Revised Edition. Dover Publications, New York (1992)
2. Murray, J.D.: On the mechanochemical theory of biological pattern formation with application to vasculogenesis. Comptes Rendus Biologies 326(2), 239–252 (2003)
3. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford Univ. Press, Oxford (1999)
4. Hastings, A.: Transient dynamics and persistence of ecological systems. Ecology Letters 4, 215–220 (2001)
5. Hastings, A.: Transients: the key to long-term ecological understanding? TRENDS in Ecology and Evolution 19(1), 39–45 (2004)
6. Prigogine, I.: The End of Certainty. Free Press, New York (1997)
7. Hamann, H.: Pattern formation as a transient phenomenon in the nonlinear dynamics of a multi-agent system. In: MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling (2009)
8. Holland, J.H.: Adaptation in Natural and Artificial Systems. Univ. Michigan Press, Ann Arbor (1975)
9. Perlin, K.: Improving noise. ACM Transactions on Graphics, Special issue: Proceedings of ACM SIGGRAPH 2002 21(3), 681–682 (2002)
10. Jones, J.: The emergence and dynamical evolution of complex transport networks from simple low-level behaviours. J. of Unconv. Comp. (2009) (accepted)
11. Haken, H.: Synergetics – an introduction. Springer, Berlin (1977)
12. Goodman, E.D.: An introduction to galopps–the genetic algorithm optimized for portability and parallelism system, release 3.2. Technical Report 96-07-01, Intelligent Systems Laboratory and Case Center for Computer-Aided Engineering and Manufacturing, Michigan State University (1996)
13. Höfer, T., Sherratt, J.A., Maini, P.K.: Dictyostelium discoideum: Cellular self-organisation in an excitable medium. Proceedings of the Royal Society London B 259(1356), 249–257 (1995)
14. Murray, J.D.: A prepattern formation mechanism for animal coat markings. J. Theor. Biol. 88, 161–199 (1981)
15. Meinhardt, H.: Models of biological pattern formation. Academic Pr., NY (1982)

Evolving Group Coordination in an N-Player Game

Enda Barrett, Enda Howley, and Jim Duggan

Department Of Information Technology
National University Of Ireland, Galway
firstname.secondname@nuigalway.ie

Abstract. The evolution of coordination is an important consideration in living systems. Throughout the natural world examples of coordination can be found. These include fish schooling, birds flocking and animals hunting in packs. This paper examines the issue of coordination and how groups can coordinate their actions in a competitive setting. A number of existing game theoretic representations of coordination have been proposed. Much of the existing research has studied two player coordination games. This paper will investigate the emergence of coordination through an n-player game. The use of signalling, communication and norms is common when attempting to address the topic of coordination, yet in this paper we will not apply any of these approaches. This paper investigates the effect of group structures on the evolution of coordination in a population of self interested individuals. The results will demonstrate the importance of these group structures when attempting to evolve coordinated actions.

1 Introduction

Research involving computer science and behavioural biology have made significant contributions relating to the area of coordination. These include new insights into aspects of flock formation [9] and fish schooling [14]. The subject of coordination is an important consideration for understanding many of these behaviours as they involve highly coordinated actions by numerous individuals. Game theoretic simulation offers us a means by which we can study the emergence of coordination. This topic of coordination has been examined widely in a number of research areas such as multi-agent systems [5], economics [10] and the social sciences [1]. A number of game representations have been commonly used to study the coordination problem such as the "Stag Hunt" game [16] [11] and the "Prisoners Dilemma".

This paper examines the evolution of strategies in an n-player coordination game. This involves studying groups of individuals coming together and interacting through a specified coordination game. The rewards they receive are based on their own choices and the choices of those in the group around them. This research aims to examine in detail the impact of groups and group sizes on the evolution of coordination. This will focus on how group structures effect the evolution of coordination and the significance of the strategies evolved. Therefore,

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 450–457, 2011.

this paper will address two research questions. Firstly, how do group structures effect the emergence of group coordination? Secondly, how do group sizes impact on the levels of coordination achieved? The following sections of this paper are structured as follows: Background Research, Simulator Design, Experimental Results and Conclusions.

2 Background Research

2.1 Coordination Games

A number of coordination games have been widely used throughout existing research. In this paper we use game theoretic simulation as a means of examining the evolution of coordination. The Stag Hunt game was first proposed by Rousseau as a representation of how two individuals might coordinate their actions when trying to hunt for their survival. The game was revisited by Skyrms et al. which has resulted in the game receiving more attention in the last number of years [12]. The game involves two players who can choose to either hunt stag or hare. A player cannot bring down a stag by themselves. They rely on the cooperation[1] of others to successfully hunt a stag. In contrast a player may bring down a hare by themselves. Choosing to hunt stag results in a richer meal but depends on the cooperation of others. Alternatively, choosing to hunt hare is less nourishing but can be hunted on one's own. The payoff matrix is shown in Table 1 and the following constraint is typically used in this game: $R > T \geq P > S$.

Table 1. The Stag Hunt

Players Choice	Stag	Hare
Stag	(R, R)	(S, T)
Hare	(T, S)	(P, P)

2.2 N-Player Coordination Game

This paper examines the evolution of coordination among groups of individuals and therefore an n-player coordination game is required. This is achieved through adapting the two player stag hunt game to an n-player game, using an order statistic game. An order statistic game [2] is one in which players must choose numbers in a certain range. Their individual payoff is a function of both their own choice and an order statistic (minimum group value). In these games each number represents a strict pareto-ranked equilibrium meaning that once a pareto optimal scenario is reached agents will not deviate from their chosen strategies. The difficulty in achieving coordination arises when we increase the number of players in the game. Hume et al. proposed a scenario where two neighbours would agree to drain a meadow, which they possess in common. He reasoned

[1] Cooperation is used throughout this paper to denote the coordination of two players.

that it was easy for each of them to know the others mind, as failure by one of them to play his part would result in the abandonment of the whole project [6]. But this becomes very difficult as the number of participants increases.

The following n-player coordination game is based on the research of Van Huyck et al. involving groups of undergraduate students. This experiment involved forming groups of 14-16 undergraduate students to play a game. During each round of the game players were asked to choose an integer between 1 and 7. Players received a payoff which was governed by their strategy and the lowest strategy held by a member in their group. No pre-play negotiation was allowed between players but after each round the minimum value strategy of the group was announced publicly [7].

Table 2. N-Player Coordination Game

		Smallest Value of X Chosen						
		7	6	5	4	3	2	1
	1	1.30	1.10	0.90	0.70	0.50	0.30	0.10
Your	2	-	1.20	1.00	0.80	0.60	0.40	0.20
choice	3	-	-	1.10	0.90	0.70	0.50	0.30
of	4	-	-	-	1.00	0.80	0.60	0.40
x	5	-	-	-	-	0.90	0.70	0.50
	6	-	-	-	-	-	0.80	0.60
	7	-	-	-	-	-	-	0.70

A payoff table (Table 2) shows the payoff each person would receive given their choice of strategy and the lowest choice by a person in their group. This table was shown to all participants and the following equation was used to generate the payoff matrix shown [7]. $a \times X_{min} - b \times X_i$ represents a players payoff equation. a represents the benefit of hunting stag, while b represents the opportunity cost of hunting hare. For the experiment a was given a value of 0.2 and b a value of 0.1. X_{min} represents the minimum choice of x in the group. A constant of 0.6 was added to ensure that the payoffs remained positive. Each student was allowed to see the payoff matrix. The aim of the experiment was to see if the students could coordinate their behaviour without communicating to receive the maximum payoff. As with the stag hunt the cost of defection (defection would arise through the choice of a low number) by a member of your group is large should you choose to hunt stag (select a high number). The results of the experiment showed that after a number of rounds there was an overall convergence to the risk adverse strategies and the results were classified as coordination failure. However Van Huyck also showed that coordination among individuals could be achieved by restricting group sizes. In his experiments he limited the group size to two and successfully achieved coordination among the participants.

2.3 Coordination and Groups

The difficulty in coordinating behaviour among self interested individuals has been highlighted numerous times. Ringelmann demonstrated this using a rope pulling exercise. It was found that as the number of individuals pulling on the rope was increased to total force exerted on the rope did not equal the sum of their individual pulls. A coordination loss occurred where participants were not pulling at the same time and the same direction [13]. Bornstein et al. showed that coordination could be achieved playing the Van Huyck game using inter-group competition. For these experiments he split the participants up into two groups with each group competing against one another. The group with the highest minimum value received the payoff in the case of a tie the payoff was divided amongst the groups [10]. Gordin et al. investigated the evolution of groups using a room painting scenario. Two separate agents a white washer and painter were required to cooperate in order to paint a room. The white washer needed to wash the wall first before it could be painted [4]. In his research involving coordination Pestelacci et al. showed that it was possible to achieve cooperation in co-evolving networks using the stag hunt [8].

3 Simulator Design

The simulations presented in this paper involve the evolution of agent strategies over successive generations. This involves a population learning through a genetic algorithm. Similar to the research outlined by Traulsen et al. the agents in the population play within groups [15]. Our population is subdivided into groups, where individuals interact only with those within their group. A game is played in each group resulting in individuals receiving a certain payoff. After each round the fittest individuals are chosen using roulette wheel selection, crossover and mutation occur and offspring are added to the same group as their parents. In the case of parents from different groups, the offspring are added randomly to one of the parents groups. As a result of this process, the fittest groups should grow at a cost to the least fit groups. Once a group reaches a critical size n it is split into two groups with probability q. We implement this process inline with the approach of Traulsen et al.

Each agent possesses a single strategy gene which contains that agent's strategy. This strategy is encoded into the gene as a bit string representing numbers in the range 1 to 7 which specify the strategy of an individual for its lifetime. Every agent belongs to a group but has no knowledge about any of the agents in its group.

Evolutionary Algorithm: In this paper we use a Genetic Algorithm (GA). The entire population is allowed to evolve once each group has played their specified number of games. Agents are allocated a fitness from these interactions, which is represented as $F_A = P_A/N_A$. The fitness of an agent A is based on the total payoffs P which that agent has received in the previous generation and the

number of games it has played N in that generation. A population of 100 individuals were used throughout these simulations. Individuals were selected using roulette wheel selection, based on their individual fitness. Elitism was applied to the population, moving the fittest E number of agents directly into the next generation. E is calculated using the elitism percentage chosen prior to running the simulation. In our simulations this elitism value was set to 5%. A cross-over rate of 85% and a mutation rate of 3% is also used. The agent population is initially dispersed randomly over 25 groups, ensuring an even distribution of agent strategies. Each agent plays 30 stag hunt games with their group peers in each generation. Offspring are created using a single point cross-over of both parents' strategy gene, with the actual cross-over point being randomly selected each time. Mutation occurred probabilistically throughout this process. Occurrences of mutation were biased towards an increase or decrease of only 1 of a possible 7 in real strategy terms. Pairings produce a single offspring and this new agent is randomly added into one of the parents' groups.

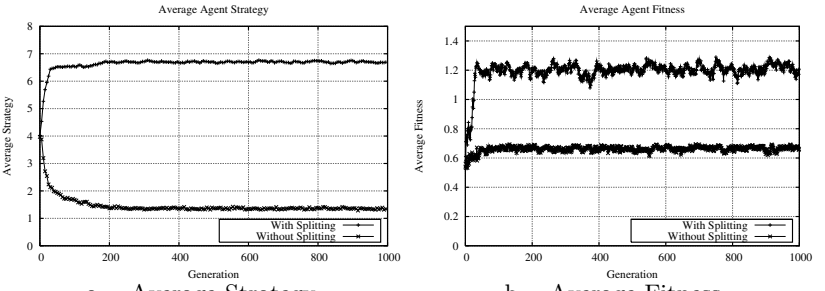
Agent Interactions: In our simulations, agents are initially assigned randomly to 25 groups. In each group all individuals participate in the coordination game. Since the strategies outlined here are not probabilistic only one round of games is required. The evolutionary algorithm then determines the representation of successive generations. An important parameter in these simulations is the size groups are allowed to grow to. An upper bound is set throughout our simulations and this is examined as an important experimental parameter. Once the upper bound has been reached, the group is split in two and agents are allocated randomly to either location.

4 Experimental Results

In this section we will present a series of experimental results examining the emergence of coordination under a number of experimental settings. The first experiment examines the impact of group splitting on the agent population. The second experiment examines the impact of group splitting by varying the upper bounds(thresholds). This splitting is similar to that which occurs in nature among various species which divide once they reach a certain size. For example, honey bees regularly swarm and divide their population [3].

Experiment 1: This experiment examines the effects of groups on the population. Fig 1 (a) shows the average strategies for agents when groups are split and not split, while Fig 1 (b) shows the average fitness.

The data shown in Fig 1 comprises a single experimental run of 1000 generations. We observe that when groups are allowed to grow without limitation the more risk adverse strategies thrive. Individuals benefit from limiting their risk through choosing a lower value strategy. This is the equivalent of choosing the hare in the traditional stag hunt game. Alternatively, when groups are split



a. Average Strategy b. Average Fitness

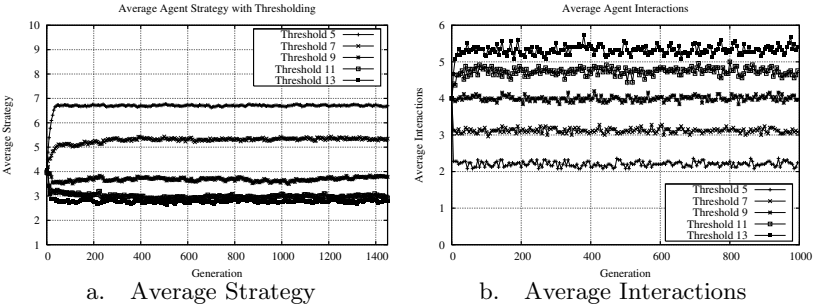
Fig. 1. Splitting Vs. No Splitting

they evolve to the cooperative payoff dominant equilibrium. This is similar to the conclusions of Traulsen et al. involving the Prisoner's Dilemma [15].

The population size is constant and therefore the growth of a single group means other group representations decrease. New offspring can only be added to a reducing set of possible groups and therefore these small numbers of groups increase in size each generation. As the number of groups grow, intergroup selections become less likely and new offspring are simply added to their parents group. Due to the nature of these games, a single low strategy agent will have a distinct advantage if groups grow to large sizes. Exploitation proves more difficult when agents are dispersed across many groups.

Experiment 2: This experiment examines the emergence of coordination when alternative group thresholds are applied. We show the effect of the group size threshold (the point at which once the group exceeds, it is split). This experiment shows data which is averaged across twenty-five runs. The other experimental parameters remained the same as in the previous experiment.

The results in Fig 2, show the effects of thresholds on the evolution of strategies with alternative group sizes and the levels of coordination achieved. Lower group thresholds allow groups to subdivide quickly and this appears to encourage



a. Average Strategy b. Average Interactions

Fig. 2. Group Thresholds

coordination. Higher thresholds have the opposite effect. Fig 2 (a) shows consistently lower levels of coordination for thresholds ranging from nine-thirteen. The reason for this is that smaller groups find it easier to maintain coordination. These small groups are more likely to avoid selfish individuals who prefer lower risk strategies. Splitting the groups once they reach the group threshold has the effect of spreading the evolved strategy throughout the entire population. Fig 2 (b) shows the levels of interactions throughout these groups. This clarifies the impact of group size on the levels of coordination evolved. The data shows the potential impact of a risk dominant strategy in populations which have higher group sizes. These higher group sizes result in higher numbers of agent interactions and thereby expose these individuals to potential exploitation due to risk dominant strategies.

5 Conclusion

This paper has presented an examination of how group structures and group sizes impact on the evolution of coordination. We adopted the game format proposed by Van Huyck et al. [7] to evaluate the importance of group structures. The group splitting mechanisms proposed by Traulsen et al. are used throughout this paper as a means of examining the impact of groups on the evolution of strategies [15].

We can conclude that coordination can be evolved through the use of group structures. Our group structures have been shown to benefit the evolution of coordination strategies and improving the fitness of the population. Earlier we proposed two research questions, the first of these related to the effect of group structures on group coordination. Our results have shown that group structures benefit the emergence of coordination through partitioning the population and allowing payoff dominant strategies to emerge. The size of these groups appears to be a significant consideration. Groups alone will not promote the emergence of coordination, but they are an important first step. Our second research question referred to the impact of group size on the strategies that emerge. Through the various threshold levels we have shown that group size has a beneficial effect on the evolution of payoff dominant strategies.

This research has clarified many of the dynamics which are implicitly represented through many tag mediated models. In this case we have provided a clear examination of the effect of group structures and their splitting throughout an agent population. As in the natural world, this process appears to be fundamental to certain populations as coordination of smaller groups is easier than larger groups. Groups need to split in order to maintain their ability to coordinate.

Acknowledgement

The authors would like to gratefully acknowledge the continued support of Science Foundation Ireland.

References

1. Chartrand, T.L., Bargh, J.A.: The chameleon effect: the perception-behavior link and social interaction. Journal of Personality and Social Psychology 76(6), 893–910 (1999)
2. Devetag, M.G.: Coordination and information in critical mass games: an experimental study. Technical report (2002)
3. Ferrari, S., Silva, M., Guarino, M., Berckmans, D.: Monitoring of swarming sounds in bee hives for early detection of the swarming period. Comput. Electron. Agric. 64(1), 72–77 (2008)
4. Gordin, M., Sen, S.: Evolving cooperative groups: Preliminary results. In: Working Papers of the AAAI 1997 Workshop on Multiagent Learning, pp. 31–35 (1997)
5. Haynes, T., Sen, S., Schoenefeld, D., Wainwright, R.: Evolving multiagent coordination strategies with genetic programming. Technical Report UTULSA-MCS-95-04, The University of Tulsa, May 31 (1995)
6. Hume, D., Selby-Bigge, L.A., Nidditch, P.H.: A Treatise of Human Nature, 2nd edn. Clarendon Press, Oxford (1978)
7. Van Huyck, J.B., Battalio, R.C., Beil, R.O.: Tacit coordination games, strategic uncertainty, and coordination failure. The American Economic Review 80(1), 234–248 (1990)
8. Pestelacci, E., Tomassini, M.: Cooperation in co-evolving networks: The prisoner's dilemma and stag-hunt games. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 539–548. Springer, Heidelberg (2008)
9. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
10. Riechmann, T.: Dynamic behavior in minimum effort coordination games - some theory of group size and inter-group competition as coordination devices. Game Theory and Information 0503010, EconWPA (March 2005)
11. Rousseau, J.: Political writings / Jean Jacques Rousseau (1915); translated by G.D.H. Cole, Camb.
12. Skyrms, B., Pemantle, R.: A dynamic model of social network formation (April 2004)
13. Steiner, I.D.: Group process and productivity. Academic Press, New York (1972)
14. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes: autonomous locomotion, perception, behavior, and learning in a simulated physical world. Artif. Life 1(4), 327–351 (1994)
15. Traulsen, A., Nowak, M.: Evolution of cooperation by multilevel selection. P. Natl. Acad. Sci. USA 103(29), 10952–10955 (2006)
16. van Rooij, R.: The stag hunt and the evolution of social structure. Studia Logica 85(1), 133–138 (2007)

Combining Different Interaction Strategies Reduces Uncertainty When Bootstrapping a Lexicon

Thomas Cederborg

VUB AI Lab - Brussels
thomas@arti.vub.ac.be

Abstract. When bootstrapping a new language, the agents in a population need to be able to agree on the meaning of the individual words. In order to do so, they need to overcome the problem of referential uncertainty, which captures the idea that the meaning of words can not realisticly be transferred directly between agents nor through the environment. One way to reduce the amount of uncertainty, is to allow the agents, based on their current knowledge of the language system and the environment, to choose the interaction script they play based on a motivational system. We show the impact of this idea through a computational model on the time needed for a population of agents to converge on a shared language system and how the motivational system allows the agents to self-regulate this process.

1 Introduction

In recent years, there has been an increasing amount of computational and mathematical models within the language game framework that investigate various aspects of the origins and evolution of language, see [1] for an introduction. In the research at the VUB AI-Lab and Sony CSL Paris we frame our models in a larger language game paradigm. In short this means all experiments consist of multiple agents that engage in pairwise interactions (language games) in a shared environment. By playing these language games the agents gradually bootstrap and maintain a communication system in order to solve a communicative task. It is important to note that there is no central control and the agents can never access the internal states of others. Different issues have already been tackled by other researchers in this field, including a detailed study of shared lexicons [2], spatial language [3], color [6], hierarchy and recursion [4] and case grammar [5].

In this research we investigate the impact of using different language game scripts for solving the problem of referential uncertainty (also known as the Gavagai problem in linguistics [14]). As an example of this problem imagine one agent describing an object, let's say a bowling ball, with the word "gavagai". Another agent, even knowing the word was used to refer to a bowling ball, still cannot with complete certainty know what part of the object was described, it could still mean; black, round, large, heavy, shiny, hollow, etc. This problem has received quite some attention in the last few years, for example [7] [8]. The key in

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 458–465, 2011.

the research presented here, is that agents autonomously choose which language game to play based on the environment and their estimates of their own and their interaction partners linguistic capabilities.

2 Experimental Setup

In the experiments we used a population of ten agents from which, at the beginning of a language game, two are randomly drawn. The context (or scene) consists of ten randomly selected objects, each composed of different features, such as shape and color. The agents perceive the scene and both build the exact same worldmodel. This simplification allows us to study several communicative problems separately from the problems arising from having different world views. The communicative task the agents need to solve is to bootstrap and align a set of form to meaning associations. These meanings however, can not only refer to the instantiated features (for example "red", "blue", "round", we shall call these descriptive words) but also to the more general dimensions (for example, "color" or "shape"). This second type of words the agents will use to ask questions, which is why we call them question words.

One of the two agents starts the game (the initiator) by first deciding what game to play. This choice is based on the scene and the agent's own lexicon, including the scores of his lexical constructions. How these scores are updated is explained in more detail below but in general words that are used in successful games will be rewarded, thus this score hints at the chance of being understood in future interactions.

The initiator can choose from among three different types of games, each having its own functional goal. The first is what we call a Guessing Game [9] [7] [8], specifically with regard to the problem of referential uncertainty. The goal of the Guessing Game is to learn the descriptive words. The interaction pattern starts with the initiator refering to one object by uttering the highest scored word for one of the objects features. The respondent either points to an object if he believes it fits the word or does nothing if he doesn't know the word or cannot map it to any object in the context. If the right object is pointed to, the initiator nods and otherwise shakes his head and points to the correct object.

Several previous experiments have examined what happens if this is the only interaction pattern available to the agents. Many different methods of analysis have been tried, both involving single word and multiple word versions [9]. We illustrate the principle of the game with an example in which the purpose of the game is to negotiate a word for red and the communicative goal is to get the other agent to point to a specific red object.

1) initiator: "red".
2) respondent: silent (does not understand the word "red").
3) initiator: points to a small triangular red object.
4) respondent: makes three hypotheses about what the word "red" means: small, triangular or red.

The other two interaction patterns both allow questions to be asked. The goal of the first Question Game (type 1) is to learn question words (i.e. words for dimensions). In this game the initiator points to an object and utters a question word (such as "color"). If the respondent is able to interpret this word, he will try to describe the object using a corresponding descriptive word, otherwise he remains silent. The initiator nods when the description is satisfactory and otherwise he shakes his head and gives the description he was hoping for. In the following example the initiator has a word for red with a score that is high enough for him to assume that the respondent knows it. The purpose of the game is to negotiate a word for the color dimension, and the communicative goal is to get the object described by the word "red".

1) initiator: points to a big round red object.
2) initiator: "color"? (assuming the word "red" is known by the other agent).
3) respondent: silent (does not understand the word "color").
4) initiator: "red".
5) respondent: learns how to describe an object when hearing "color".

In question game type 2 the initiator uses a question word he thinks the respondent knows to ask for the name of a feature. In the following game the initiator has a word for color with a high enough score to assume that the respondent will know it. The purpose of the game is to negotiate a word for yellow, and the communicative goal is to get the object described by the word the respondent has for yellow.

1) initiator: points to a big triangular yellow object.
2) initiator: "color"? (assuming the word "color" is known by the other agent).
3) respondent: "yellow".
4) initiator: nods (does not have a word for it).
5) initiator: Learns the word "yellow" (knowing it is not referring to big or triangular).

The question games will obviously result in incorrect learning if the assumptions made by the agents are incorrect. If the respondent in the second example is sure that "red" means round, he will also conclude that "color" must mean shape. The initiator of the third example will also make incorrect inferences if his assumption about the respondent's knowledge of "color" is wrong. The agents will have to hear a descriptive word being used by other agents (describing objects that fit with what the agent thinks the word mean) before feeling confident enough to use it in a question game. With every word there is associated a score with a value between 0 and 1 of the same type as described in [9].

The motivation of the agents is to increase the expected communicative success in situations where they are not allowed to choose the topic of discussion. This can be seen as them having some reason, other than language learning, to describe an object. The different types of games they can play receive their utility from this goal. The communicative goals of all the games they play (for example the communicative goal: getting an other agent to describe a red object using color) should be seen as subgoals of this. When deciding what game to play and

what the topic should be the agents try to find an appropriate challenge level by simultaneously trying to avoid anxiety and boredom. This approach draws inspiration from psychology and the concept of flow [12], a mental state where the challenge (and therefore the experience) of some task is optimal. A typical example is that of a mountain climber who has managed to find a mountain that is very difficult and challenging (and thus not boring) but that does not contain any obstacles that are so difficult that he does not know how to start dealing with them (something that would have produced anxiety) and has been explored before in the context of language learning, in which it was called the autotelic principle [10].

The utility U of playing a specific game about a specific feature is assigned by a utility function and the game with the highest utility will be played. We will call the score of the best question construction for the type of the feature Sc and the score of the best descriptive construction Sd.

For the guessing game the agents have no help other than pointing (and thus have no good way of communicating what they were actually describing). They are thus always anxious in this game and always prefer to talk about the feature they know the best as this is their only way of reducing anxiety within the guessing game framework. Thus the utility of playing a guessing game about a given feature is simply the score of the highest scoring construction with the same value as that feature. We get $U = Sd$.

For question game type 1 the uncertainty and thus the anxiety comes from 2 sources, the uncertainty of the question construction used and the uncertainty about the descriptive construction used. We get:

$$U = ((Sc^3) * 32) * (1/(1 - Sd)). \tag{1}$$

Here the term $(1/(1 - Sd))$ can be seen as confidence in the descriptive construction used to learn the question construction. If no question construction is available the utility function is set to that corresponding to a score of 0.5.

For question game type 2 there are 2 competing factors, the will to learn a descriptive construction that should increase with a low score for this construction and a fear of failure as a result of the question construction. We get:

$$U = (2 - Sd) * (1/(1 - Sc)). \tag{2}$$

Here the first term is from wanting to learn descriptive words not known (avoiding boredom) and the second term anxiously avoiding to use bad question constructions to learn them. When using color to learn green a high score of color (to reduce risk of failing due to the other agent not understanding the question) is preferred and a low score of green is preferred (wanting to learn what he does not already know well). This is a good example of how the will to avoid anxiety can conflict with the will do avoid boredom. If a game is found that manages to avoid both of these (finding a question word that is sure to be understood and is useful for learning the name of a feature that is completely unknown) we can see this as the agents having achieved flow in the sense of [12]. U is set to 0 if boredom is very high (the word that can be learnt is very well known)

or if anxiety is to high, for example asking a question that you do not know the answer to using a question word that other agent almost certainly will not understand.

When an agent has completed a language game he will analyze what has happened and can update his language in three ways; he can adopt a construction (when a new word is heard), he can change the score of a construction (based on how likely it is that other agents share this construction) and finally he can delete a construction (when the score is below a threshold of 0.5) [1]. When a descriptive word is used about an object the construction with its form is increased if it's meaning matches the object described and decreased if it does not match. If the score of one descriptive word is increased the score of constructions that are it's homonyms or synonyms are decreased. This is called lateral inhibition and is a standard feature in language game experiments, see [11] for an early implementation. When a new descriptive construction is heard the agent will adopt multiple constructions if he is not sure what the word means, one for every possible meaning. If during a question game type 2 the agent knows exactly what feature the new word is describing he only needs to adopt one construction.

The scores of question constructions are updated in similar ways. If a question game (type 1 or 2) is successful the score of the question construction that was used is increased as it is almost impossible to succeed in such a game if the meaning of the question construction is not shared amongst the agents. If a question game type 1 fails the score of both the question construction and the descriptive construction used is decreased. Decreasing the score of the descriptive construction helps the agents realize that they are moving to fast from guessing to question games. This is important as the agents are autonomously regulating their own progress and moving to fast can seriously damage the language they develop. Using "red" to negotiate a word for color is only useful if all agents understand "red" the same way. If they do not understand its meaning the same way but try to use it anyway the question games will keep failing, the score of "red" will decrease and the agents will stop playing the question game (until they have settled on the meaning of "red").

3 Results

There are striking patterns in how the agents choose to play games. In the beginning they always play guessing games as the other games require some prerequisites to play. When they start seeing features that they have a word for (invented or heard) they will focus on this feature. The feedback of wanting to talk about what you know and learning the words others talk about soon makes them talk about some features more than others. The behavior will then be even more coordinated as they will attempt to find a name for color or shape and now they have similar vocabularies and thus they are all more or less able to start naming the same things (if all know red they can all learn color but if

[1] I will not go through the details of every update rule for every possible situation, only list the most important cases.

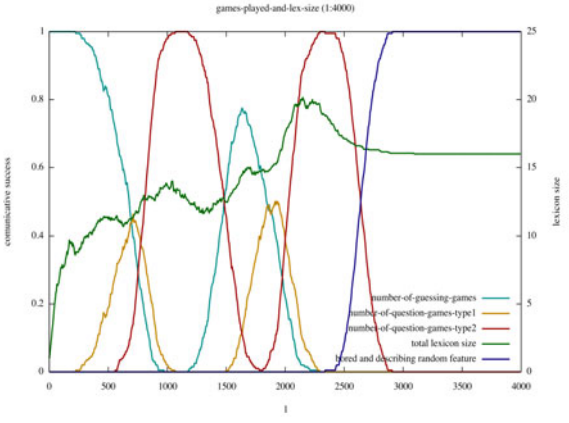


Fig. 1. The figure shows the results of an experiment with 2 different types of features. A pattern can be seen of first guessing games (light blue) followed by learning a question word (yellow) followed by using the question word for further learning (red). The pattern is then repeated when the question word can not be used for learning new things any more ("shape" can be used for learning "round", "square" etc but when all shapes are named a new question word must be invented to move forward). Finally the agents get bored (dark blue) when they are presented with a scene with only features that hey have good names for. The average lexicon size is shown in green.

they all know round they can all learn shape). When they have a word for color they start using it to learn the names of all other colors and now they are very coordinated as can be seen in figure 1. As they learn all the color words they get bored and start the whole process again with for example shape and so on resulting in the wave pattern seen.

To measure how successful an experiment is we need to introduce a formal measure of how good the language they have developed is. Since the agents themselves not only chose the game to play and the topic to talk about, but many times do this in a way that reduces risk of failure (they often choose games that are as easy as possible), a measure of how often individual games succeed is not a very good measure of performance. The communicative goals are only subgoals of learning a language. An agent that makes incorrect approximations about how useful a communicative subgoal is in learning a language can systematically cause it to play very easy games that are very successful but that does not teach it anything. It will have its subgoals met more often than an agent who chooses games where learning takes place (but that fails sometimes) but the former agent should be seen as less successful than the latter.

I introduce lexical coherence as the measure of success for the remainder of this paper. This is defined as the probability that when 2 agents are drawn at random and a feature of the world is picked at random they will both have at least one construction with this feature at the meaning pole and that the highest scoring such constructions for the two agents will have the same form. Put in

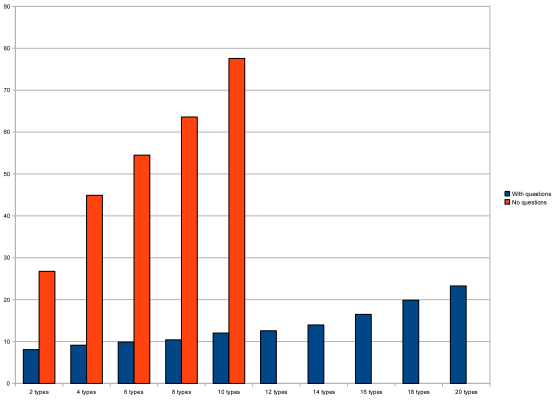


Fig. 2. The blue columns are convergence times for agents asking questions and the red are convergence times for agents playing only the guessing game. All experiments involve 10 agents. Convergence times are measured in number of interactions per feature and agent to reach 98% coherence. It is clear that the questions give the agents a significant boost in performance.

another, and less formal way, it is the probability that two agents will prefer the same word for a feature.

We can get a good overview of how performance improves in figure 2. The convergence times of the agents asking questions are shown in blue and those not asking questions are shown in red. A comparison is made in five different worlds with 2, 4, 6, 8 and 10 types of features (2 types would correspond to a world with shapes and colors, 4 would be a world with shapes, colors, texture and size, etc). An average has been taken over 20 runs in the case of guessing games. For the cases being compared (2,4,6,8 and 10 types) 50 runs have been averaged in the question games. For the question games even more complex worlds have been tested up to 20 different types. For these five experiments fewer runs have been used to average the results (between 5 and 20).

4 Conclusion

The problem of referential uncertainty is an important obstacle when bootstrapping a language in a population of agents from scratch and has been investigated by several different computational approaches before. We have introduced another approach on how this uncertainty can be reduced, by giving increased autonomy to the agents to choose among different interaction scripts based on a motivational system. We have shown that such a system allows the population to achieve significant improvements in the time needed to converge on a language system. It was also demonstrated that agents can self-regulate their interactions and decide when they are ready to move from one type of interaction to another even if they have no global coordination, no way of directly observing the abilities of other agents and only partial information about the interaction history

of the population. This suggests that when investigating a linguistic phenomena where different learning stages are required, it might not be necessary for the experimenter to guide the experiment using global information or information that is private to the agents.

References

1. Steels, L.: Evolving grounded communication for robots. Trends in Cognitive Science 7, 308–312 (2003)
2. Baronchelli, A., Loreto, V., Steels, L.: In-depth analysis of the naming game dynamics: the homogeneous mixing case. Int. J. Mod. Phys. C 19, 785 (2008)
3. Steels, L., Loetzsch, M.: Perspective alignment in spatial language. In: Coventry, K.R., Tenbrink, T., Bateman, J.A. (eds.) Spatial Language and Dialogue. Oxford University Press, Oxford (2007) (to appear)
4. De Beule, J.: The emergence of compositionality, hierarchy and recursion in peer-to-peer interactions. In: Smith, A.D.M., Smith, K., Ferrer-i-Cancho, R. (eds.) Proceedings of the 7th International Conference on the Evolution of Language, pp. 75–82. World Scientific, Singapore (2008)
5. van Trijp, R.: The emergence of semantic roles in Fuid construction grammar. In: Smith, A.D.M., Smith, K., Ferrer-i-Cancho, R. (eds.) Proceedings of the 7th International Conference on the Evolution of Language, pp. 346–353. World Scientific, Singapore (2008)
6. Steels, L., Belpaeme, T.: Coordinating perceptually grounded categories through language: A case study for colour. Behavioral and Brain Sciences 28(4), 469–489 (2005)
7. Wellens, P., Loetzsch, M., Steels, L.: Flexible word meaning in embodied agents. Connection Science 20, 173–191 (2008)
8. Smith, A.D.M.: The inferential transmission of language. Adaptive Behavior 13(4), 311–324 (2005)
9. De Beule, J., De Vylder, B., Belpaeme, T.: A cross-situational learning algorithm for damping homonymy in the guessing game. In: Rocha, L.M., et al. (eds.) Artificial Life X, pp. 466–472. MIT Press, Cambridge (2006)
10. Steels, L., Wellens, P.: Scaffolding language emergence using the autotelic principle. In: IEEE Symposium on Artificial Life, pp. 325–332 (2007)
11. Steels, L., Kaplan, F.: Spontaneous lexicon change. In: COLING-ACL 1998, ACL, Montreal, pp. 1243–1249 (1998)
12. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper Perennial (1991)
13. Steels, L.: The autotelic principle. In: Iida, F., Pfeifer, R., Steels, L., Kuniyoshi, Y. (eds.) Embodied Artificial Intelligence. LNCS (LNAI), vol. 3139, pp. 231–242. Springer, Heidelberg (2004)
14. Quine, W.: Word and Object. MIT Press, Cambridge (1960)

A Chemical Model of the Naming Game

Joachim De Beule

Artificial Intelligence Lab, Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussel
joachim@arti.vub.ac.be
http://arti.vub.ac.be/~joachim

Abstract. A key feature of many biological distributed systems is that they have the capacity to behave in highly coordinated ways. In the domain of language, coordination dynamics have been studied within the framework of language games. As yet however, a fundamental understanding that goes beyond the simplest cases is still missing.

In this paper, a novel approach is proposed for investigating coordination problems. The approach is illustrated for a simple but well studied case called the naming game. It is also argued that the proposed approach provides a good starting point for tackling more complex coordination problems as well.

Keywords: Coordination, Semiotic Dynamics, Agent Response Analysis, Artificial Chemistry, The Naming Game, Idiotypic Networks.

1 Introduction

A key feature of many biological distributed systems is that they have the capacity to behave in collective and highly coordinated ways. In the domain of language, such coordination dynamics have already extensively been studied using the concept of language games[10]. In a language game, a population of agents (language users) needs to bootstrap a 'language' by engaging in pairwise interactions. For example, in the naming game they have to agree upon a name for an object.

In more complex games, agents would benefit from using more complex encodings that go beyond simple naming and involve syntax, similar to natural languages inducing a conventional and multi-levelled mapping between hierarchically structured meanings and forms [6,9], or to the immune system which is capable of responding appropriately in a virtually infinite number of different situations [7].

Although (especially in recent years) the naming game has become very well understood (see e.g. [2]), as yet only very few results exist that go beyond it and a fundamental understanding of the dynamics involving the emergence and evolution of syntax is still lacking.

In this paper, I propose to model agents as artificial cellular organisms interacting with their environment through the absorption and secretion of artificial

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 466–473, 2011.

chemical substances (henceforth called species) and in turn modelled as continuous stirred flow tank reactors with entrapped species [3]. Within a cell/reactor, species interact themselves and new species are formed according to a reaction network corresponding to the learning and entrenchment of linguistic constructions in more traditional agents.

This approach lends itself particularly well for performing an *agent response analysis* [11,4], which allows do draw conclusions about the behavior of a population of agents on the basis of a single agent only. In the following section, this will be illustrated for the case of the naming game. A reactor agent solving the naming game will be defined and a response analysis will be performed on it. This will lead to the identification of a design principle for synthesizing artificial agents solving more complex coordination problems, as will be discussed in the final section of the paper.

2 A Reactor Agent Solving the Naming Game

Consider the following game. Every round two (random) players enter a room. Both players write a list of names on a paper, and papers are exchanged before they leave the room again. The game ends if all players consistently write down one and the same name. This game is called the naming game.

In traditional approaches, players are called agents. They typically keep a memory of names, possibly with a preference measure (a score), and apply one or the other lateral inhibition mechanism after each game: names in the other player's list are promoted at the expense of other names. It is not primarily clear however how this approach scales to more complex languages.

Now consider the chemical reactor tank in Figure 1(b) as a model for a cellular organism playing the naming game (Figure 1(a).) On the left side, the reactor is fed with a continues supply of name species corresponding to the other player's

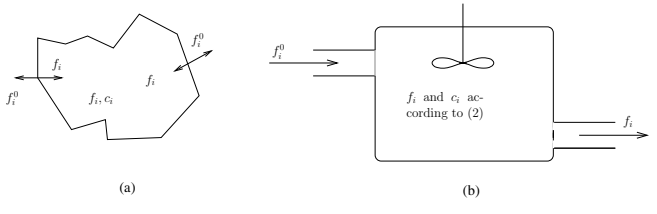


Fig. 1. A cellular agent (a), modelled as an isothermal homogeneous continuous flow stirred tank reactor (b). The reactor is supplied with a continues feed of form species (names), with molar concentrations f_i^0. Inside the reactor/cell new species c_i are formed according to the idiotypic reaction network shown in Figure 2. Forms are also continuously extracted from the reactor/cell (with f_i the molar concentrations of form species i both in the reactor/cell and in the outflow.) The new species remain entrapped in the cell.

$$f_i \xrightarrow{k_1} c_i$$
$$c_i \xrightarrow{k_2} f_i$$
$$c_j + f_i \xrightarrow{k_3} c_i + f_i$$

Fig. 2. An artificial chemical reaction network. According to the first two reactions, form species f_i trigger the existence of an anti-idiotypic species c_i and vice versa. Setting the reaction rate $k_2 = 1 + f_i$ renders species f_i autocatalytic.

lists of names (f_i^0).[1] More formally, f_i^0 denotes the molar concentration of names of type i in the influx (where $i = 1..n_f$ and n_f is the number of different names). With

$$\sum_i^{n_f} f_i^0 \equiv 1 \ , \tag{1}$$

which is arbitrary, f_i^0 corresponds to the frequency with which other players use name i. Similarly, f_i corresponds to the frequency with which the agent under investigation uses name i in a game. Because of the stirred tank hypothesis, it also corresponds to the molar concentration of name species i withing the agent.

Inside the agent, each form f_i lives in a co-existential balance with an anti-idiotypic species c_i according to the first two reactions in Figure 2. These thereby form a kind of memory similar to what was already proposed in [7].

In this paper, it is in addition assumed that the equilibrium balance ratio c_i/f_i depends on the amount of available species (i.e. c_i and/or f_i). This can be accomplished in several ways. For example, assume that $k_1 = k_3 = 1$ and that $k2 = 1 + f_i$. The last condition renders species f_i autocatalytic. It makes the equilibrium ratio's $c_i/f_i = k_1/k_2$ induced by the first two idiotypic reactions in Figure 2 depend on the amount of available f_i. This will turn out to be crucial when considering the interplay with the third reaction.

To see this, let us turn back to the complete reactor model. Define

$$\sigma_f \equiv \sum_i f_i \quad ; \quad \sigma_c \equiv \sum_i c_i \ , \tag{2}$$

and assume a mass-action kinetics. If form species are supplied and extracted at a volumetric flow rate ρ_f (with dimension volume/time) and with

$$c_i^* \equiv f_i/(1 + f_i) \ , \tag{3}$$

we arrive at the following system of differential equations describing the agent state change over time when confronted with a particular external (population) behavior f_i^0:

[1] I use f to denote names to emphasize the generality of the approach: their is no restriction on the sort of species in the influx, and in more complex games instead of names there will also be other 'elementary' as well as more complex 'parts of form'.

$$\dot{f}_i = \rho_f(f_i^0 - f_i) - (1 + f_i)(c_i^* - c_i) \ ,$$
$$\dot{c}_i = (1 + f_i)(c_i^* - c_i) + f_i\sigma_f(\frac{\sigma_c}{\sigma_f} - \frac{c_i}{f_i}) \ . \tag{4}$$

Note that because of the reactor hypothesis we can be certain that $f_i > 0$ if $\rho_f > 0$ and $f_i^0 > 0$. The first term in $\dot{f}_i$ represents the in and out flux of form species. It drives the agent to adopt the population behavior (f_i^0). The second term is opposite to the first term in $\dot{c}_i$. These induce the abundance-dependent equilibrium ratio's as, by definition:

$$(c_i = c_i^*) \Rightarrow (c_i/f_i = 1/(1 + f_i)) \ . \tag{5}$$

In words: less abundant forms will have higher equilibrium ratios. The final term however drives all ratios to become equal to σ_c/σ_f. Furthermore, it does so by converting species with higher ratios to species with lower ratios. In other words: *by converting already less abundant forms to already proliferating forms.* This interplay makes that this agent solves the naming game as will be shown in the next section.

3 Agent Response Analysis

The idea to investigate coordination problems through an agent response analysis was already introduced in [11] and [4]. In this section it will be shown how this method also applies naturally to the reactor agent defined in the previous section.

For this, and with $\Re^+$ denoting all positive real numbers, define the *agent state space* $Q = (\Re^+)^{2*n_f}$ as the set of all possible agent states $q = \langle f_i, c_i \rangle$, and let the *behavior space* $F = [0, 1]^{n_f}$ be the set of all distributions over forms (i.e. behaviors). Then the agent's *behavior function* becomes:

$$f : Q \to F : \langle f_i, c_i \rangle \mapsto f_i/\sigma_f \ . \tag{6}$$

Furthermore, the agent's *transition function* becomes a function:

$$\delta : Q \times F \times \Re^+ \to Q \ , \tag{7}$$

where $q(t) = \delta(q(0), f^0, t)$ is the solution to (4) when integrated over a time t and starting from an initial state $q(0)$ at time 0.

Finally, an agent's *response behavior* is defined as the agent's behavior when confronted with a constant population behavior for a very long time. If an agent is ergodic, then this limiting behavior will be independent of the initial state $q(0)$ and converge to a steady state under all circumstances. In this case it makes sense to define the agent's *response function*:

$$\phi : F \to F : f^0 \mapsto lim_{t\to\infty} f(\delta(q(0), f^0, t)) \ . \tag{8}$$

It maps a (constant) external population behavior f^0 to the agent's unique limiting behavior induced by it. Now define

$$f^* = c^* \frac{\sigma_f}{\sigma_c} \ , \tag{9}$$

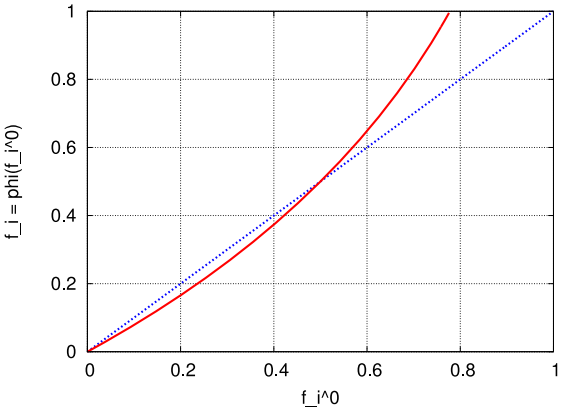


Fig. 3. The curved line represents the response function of the cellular agent defined in section 2 for values $\sigma_f = \sigma_{f^0} = 1$ and $f^* = 0.5$ (see text.). It determines the agent's outflow behavior $f_i = \phi(f_i^0)$ (Y-axis) given the inflow behavior f_i^0 (X-axis) and relative to the invariant frequency f^*. Input frequencies below f^* are dampened while those above it are amplified.

with

$$c^* = f^*/(1 + f^*) \ . \tag{10}$$

Thus, f^* represents the form concentration in the influx that occurs with the same concentration in the outflux and corresponds to the equilibrium form concentration of the system (4) under the condition that $f_i = f_i^0$.[2] Solving for σ_c and substituting the result into equations (4) with $\dot{f}_i$ and $\dot{c}_i$ set to zero results in an expression for the agent's response function ϕ in terms of f_i^0 and containing the parameters ρ_f, σ_{f^0} and f^*. It is shown in Figure 3 for $\sigma_{f^0} = 1$ and $\rho_f = f^* = 0.5$ together with the identity relation $f_i = f_i^0$.

As can be seen, inflow frequencies below the invariant frequency f^* are dampened while those above it are amplified. This suggests that if this agent were to interact with other agents like itself, they would gradually converge to a state in which only one form remains, thus solving the naming game. In particular this holds if the agent were to interact *with itself* (i.e. when $f_i^0 = f_i$ or, equivalently, $\rho_f = 0$). In [4] it is suggested how, under mean field conditions, the dynamics induced by such a closed system indeed correspond to the average population dynamics.

This can be understood as follows. Consider an agent randomly interacting with other agents in the population at times $k = 1, 2, ...$ Let $q(k)$ and $f^0(k)$

[2] Note that, because of the reactor hypothesis, in equilibrium it will hold that

$$\sigma_f = \sum_i^{n_f} f_i^0 \equiv \sigma_{f^0} \ , \tag{11}$$

and since then also $\sigma_c + \sigma_f$ is constant, f^* is entirely determined by the input distribution f^0.

represent the agent's state and the average population behavior at time k respectively. Every interaction the agent is stochastically influenced by the population behavior and vice versa. With f and δ the agent's behavior and transition functions, this results in the following set of stochastic difference equations:

$$f^0(k+1) = (1-\beta)f^0(k) + \beta f(q(k)) \quad ; \quad q(k+1) = \delta(q(k), f^0(k)) \ , \tag{12}$$

with $\beta \in [0,1]$ a constant parameterizing the degree of influence an agent has on the population. For large populations, β will be relatively small, and f^0 will remain relatively constant over a large number of interactions, meaning that the agent's behavior will approach it's response behavior $\phi(f^0)$. If we now define $f^0(k) = f^0(k\Delta t) = f^0(k)$ and let $\Delta\beta \to 0$ with $\frac{\beta}{\Delta t} = \alpha$, a constant, then the following ordinary differential equation is obtained:

$$\frac{d}{dt} f^0 = \alpha(\phi(f^0) - f^0) \ , \tag{13}$$

relating the evolution of the average population behavior f^0 to that of the fixed points of the response function ϕ of the agents constituting the population. In our case these correspond to the equilibrium states of the closed reactor system (i.e. equations (4) with $\rho_f = 0$. The top of Figure 4 shows the evolution of such a system in case of $n_f = 100$ forms. As predicted, only one form type remains in the end. The bottom graph shows the corresponding evolution of the coherence and the synonymy (scaled by the number of form types). With $w_i = f_i/\sigma_f$, these are defined respectively as:

$$\text{Coh(t)} = \sum_{i=1} (w_i(t))^2 \quad \text{and} \quad \text{Syn(t)} = \exp(\sum_{i=1}^{n_f} -w_i(t)\log(w_i(t))) \tag{14}$$

and can be related to the communicative success and the number of words in the population. As can be seen, a very sharp transition occurs around $t = 850$, in accordance with previous findings resulting from multi-agent simulations (see e.g. [1].)

4 Discussion and Conclusion

In this paper, it was shown how a simple coordination problem like the naming game can conveniently be studied by combining concepts from (artificial) chemical reaction network theory and systems biology with earlier notable attempts to systematize the investigation of coordination problems through an agent response analysis.

I argue that this approach is also particularly well suited for investigating more complex coordination problems. The response analysis not only suggests a relation between the behavior of single agents and that of a population, it also points towards an intuitive agent design principle: an agent will only be fit for a particular coordination problem if it can solve it when interacting with itself.

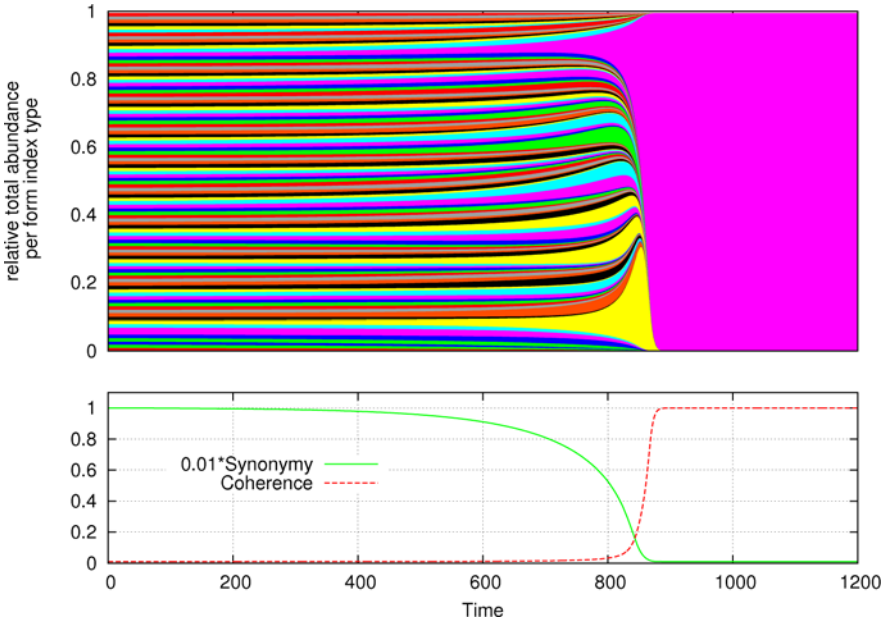


Fig. 4. Evolution of a system governed by equations (4) with $\rho_f = 0$ and $n_f = 100$ and starting from a random initial state. The top graph shows the relative total abundances $(f_i + c_i)/(\sigma_f + \sigma_c)$ for all form index types $i = 1..100$. The bottom graph shows the corresponding evolution of the coherence and (scaled) synonymy of the system, roughly corresponding to the communicative success and the number of words in more traditional setups.

By modelling agents as continues flow reactor tanks this condition is easily checked by feeding back an agent its own outflow and performing a stability analysis on the induced set of differential equations. In other words, designing an agent boils down to finding a reaction network inducing a dynamics with the desired properties according to the coordination problem under investigation.

Moreover, for this we will now have at our disposal a wealth of powerful insights about reaction networks coming from artificial chemistry and chemical reaction network theory (CRNt) [5,8]. For example, the question whether a reaction network supports multiple positive stable equilibria is particularly well studied in CRNt. Because coordination problems typically require that agents are able to escape from incompatible (sub-optimal) configurations, I predict that this will be of importance for more complex coordination problems.

We will also be able to draw upon (and perhaps even help understand) an increasing amount of available data and insights from systems biology and related fields regarding biological reaction networks. As a first example, I have shown how an idiotypic reaction mechanism, first identified within the context of the immune system, induces a sharp transition from a random and incoherent state to a highly coordinated one thus solving the naming game. I predict that other

well studied mechanisms like covalent modification and phosphorylase will also turn out to be useful for tackling more complex coordination problems.[3]

Acknowledgements. This work was partly funded by the ComplexDis European sixth framework project (FP6-2005-NEST-PATH). The author would like to thank Luc Steels, Peter Dittrich, Karel van Acoleyen and Bart De Vylder for useful comments.

References

1. Baronchelli, A., Felici, M., Caglioti, E., Loreto, V., Steels, L.: Sharp transition towards shared vocabularies in multi-agent systems. J. Stat. Mech., P06014 (2006)
2. Baronchelli, A., Loreto, V.: In-depth analysis of the naming game dynamics: The homogeneous mixing case. International Journal of Modern Physics C 19(5), 785–812 (2008)
3. Craciun, G., Feinberg, M.: Multiple equilibria in complex chemical reaction networks: Extensions to entrapped species models. Systems Biology 153(4), 179–186 (2006)
4. De Vylder, B.: The Evolution of Conventions in Multi-Agent Systems. PhD thesis, VUB Artificial Intelligence Lab (2007)
5. Feinberg, M.: Lectures on chemical reaction networks. Lectures given at the Mathematics Research Center of the University of Wisconsin-Madison (1979), http://www.che.eng.ohio-state.edu/FEINBERG/LecturesOnReactionNetworks/
6. The Five Graces Group. Language is a complex adaptive system. DOI: SFI-WP 08-12-047, SFI Working Papers (2008)
7. Jerne, N.: The generative grammar of the immune system. Bioscience Reports 5(6), 439–451 (1985) (Nobel Lecture Note)
8. Matsumaru, N., Hinze, T., Dittrich, P.: Organization-oriented chemical programming for distributed artefacts (2009) (to appear)
9. Steels, L.: Language as a complex adaptive system. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., Schwefel, H.-P. (eds.) PPSN 2000. LNCS, vol. 1917. Springer, Heidelberg (2000)
10. Steels, L.: Language games for autonomous robots. IEEE Intelligent Systems, 17–22 (September-October 2001)
11. De Vylder, B.: Coordinated communication, a dynamical systems perspective. In: Proceedings of the European Conference on Complex Systems, ECCS 2006 (2006)

[3] In fact, in a subsequent paper, I will show how the introduction of domains akin to protein domains allows to decompose a more complex coordination problem called the guessing game into several instances of the naming game.

Evolving Virtual Fireflies

David B. Knoester and Philip K. McKinley

Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824, USA
{dk,mckinley}@cse.msu.edu

Abstract. In this paper, we present a study in the evolution of cooperative behavior, specifically synchronization, through digital evolution and multilevel selection. In digital evolution, a population of self-replicating computer programs exists in a user-defined computational environment and is subject to instruction-level mutations and natural selection. Multilevel selection links the survival of the individual to the survival of its group, thus encouraging cooperation. Previous approaches to *designing* synchronization algorithms have taken inspiration from nature: In the well-known firefly model, the only form of communication between agents is in the form of "flash" messages among neighbors. Here we demonstrate that populations of digital organisms, provided with a similar mechanism and minimal information about their environment, are capable of *evolving* algorithms for synchronization, and that the evolved behaviors are robust to message loss. Moreover, analysis of the dominant genome reveals that the evolved solution utilizes an adaptive frequency strategy strikingly similar to that observed in fireflies.

Keywords: evolutionary computation, digital evolution, synchronization, self-organization, cooperative behavior, distributed algorithm.

1 Introduction

The natural world is replete with organisms that exhibit cooperative behaviors of varying complexity. Some of these cooperative behaviors exhibit *synchrony*. For example, honey bees synchronize their activity cycles [1], fiddler crabs synchronously wave their oversized claw [2], and ants synchronize their alarm drumming [3]. One of the more striking examples of synchrony in the natural world is the coordinated flashing of male fireflies. In some parts of Southeast Asia, these fireflies synchronize their flashes to a common period and phase over a distance of many miles [4]. Researchers have developed several mathematical models of this behavior, among them the pulse-coupled oscillator model of Mirollo and Strogatz [5] and the adaptive frequency model of Ermentrout [6]. Such models enable the design of *biomimetic* synchronization algorithms. For example, Babaoglu et al. [7] leveraged the Ermentrout model to develop a heartbeat synchronization algorithm for large overlay networks, facilitating coordination of collective tasks among network nodes.

In the study reported here, we used AVIDA [8], a digital evolution platform closely associated with artificial life, to explore the evolution of synchronization behavior.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 474–481, 2011.

In AVIDA, a population of self-replicating computer programs (digital organisms) exists in a user-defined environment and is subject to mutation and natural selection. We extended AVIDA with a small set of instructions that enable digital organisms to transmit and receive virtual "flash" messages. We also defined multilevel selection criteria that reward groups of digital organisms for exhibiting synchronization behavior. In experiments, the AVIDA populations evolved the ability to synchronize very quickly from arbitrary initial states. Analysis of the dominant genome revealed that the solution utilizes an adaptive frequency strategy remarkably similar to that of the Ermentrout model. While other studies have investigated many aspects of the evolution of cooperative behavior [9,10,11], including synchrony [12], the main contribution of this work is to demonstrate the *de novo* evolution of a cooperative behavior for synchronization. Moreover, our experiments show that this synchronization behavior evolves even in the presence of significant loss rates of virtual flashes, suggesting that an adaptive frequency mechanism is an important element in resiliency to environmental interference.

2 Research Platform

Digital evolution [13] is a form of evolutionary computation originally developed to study evolution in biology. AVIDA [8], a platform for digital evolution, is well-suited for studies of cooperative behavior, and has previously been used in artificial life studies on the evolution of cooperative communication behaviors [14] and adaptive sleep response [15].

Figure 1(a) depicts an AVIDA population and the structure of an individual organism. Each digital organism comprises a circular list of instructions (its *genome*) and a virtual CPU, and exists in a common virtual environment. The virtual CPU contains three general-purpose registers (AX, BX, CX), two stacks, and a number of *heads* (pointers to instructions in the genome), which can be manipulated for execution-flow control. Within their environment, organisms execute the instructions in their genomes, and the particular instructions that are executed determine the organism's behavior (its *phenotype*). Instructions within an organism's genome are similar in appearance and functionality to traditional assembly language instructions. These instructions enable an organism to perform simple mathematical operations, such as addition, multiplication, and bit-shifts; to manipulate the position of heads within their genome; to sense and change properties of the environment; and to communicate with neighboring organisms. Instructions within AVIDA can also have different costs in terms of virtual CPU cycles. For example, a simple addition may cost only one cycle, while broadcasting a message may cost 20 cycles. New instructions implemented for this study are summarized in Table 1. Of particular relevance is the flash instruction, which broadcasts a message to each of the calling organism's neighbors within the virtual environment. Organisms can retrieve information about any received flashes via the if-recvd-flash, flash-info, and flash-info-b instructions.

Many approaches to the evolution of cooperation in non-biological systems involve *multilevel selection*, where selection not only acts on individuals, but also on the groups to which the individuals belong. AVIDA provides a framework for multilevel selection called CompeteDemes, which enables the periodic replication and competition

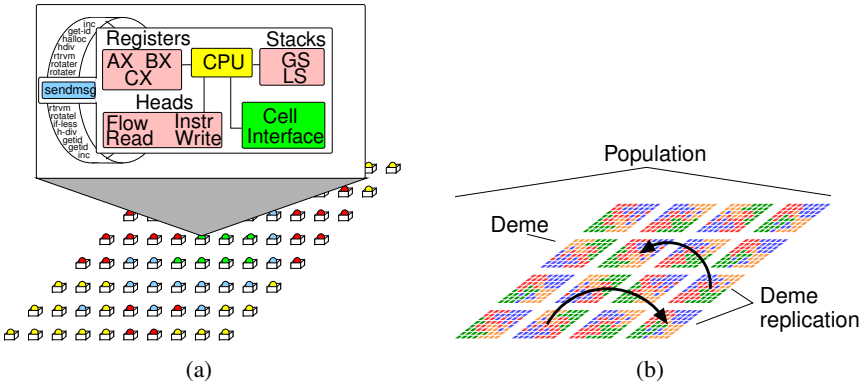


Fig. 1. Elements of the AVIDA platform: (a) an AVIDA population containing multiple genomes (bottom) and the structure of an individual organism (top); (b) depiction of an AVIDA population divided into 16 demes. When a deme replicates, it replaces a randomly selected target deme.

among *demes*. In AVIDA, a deme is an isolated subpopulation of organisms. In Figure 1(b), we see a population divided into 16 demes. During the execution of an AVIDA experiment, the CompeteDemes framework periodically calculates the fitness of each deme via a user-defined fitness function. This fitness function takes as input a single deme, and produces the fitness of that deme (a floating-point number) as output. Using the resulting array of fitness values, the CompeteDemes framework then performs fitness-proportional selection, preferentially replicating those demes with higher fitness. For example, one may define a fitness function based on completing a cooperative task. Over time, the CompeteDemes framework will then preferentially replicate those demes that are more successful than others.

Table 1. New AVIDA virtual CPU instructions implemented for this study. All instructions are equally likely to be selected as targets for mutation.

Instruction	Description
flash	Broadcasts a "flash" message to caller's neighbors, with a configurable loss rate.
if-recvd-flash	If the caller has received a flash from any of its neighbors, execute the subsequent instruction. Otherwise, skip the subsequent instruction.
flash-info	If the caller has ever received a flash, set BX to 1 and CX to the number of cycles since that flash was received. Otherwise, set BX and CX to 0.
flash-info-b	If the caller has ever received a flash, set BX to 1; do not modify CX.
hard-reset	Reset the state of the virtual CPU to the organism's "birth" state. All registers are zeroed out and heads are reset, including the flash timer and cycle counter.
get-cycles	Set BX to the number of virtual CPU cycles since either the organism was born or the last time hard-reset was called, whichever is most recent.

To encourage the evolution of cooperation, we also employed *digital germlines* [16], a framework that provides a single common genetic ancestry for all organisms within

a deme. In AVIDA, the germline is a genome attached to a deme, rather than to an individual. Although individuals within a deme can self-replicate, mutations occur only along the germline, and then only during deme replication. When a deme replicates (the arrows in Figure 1(b)), the germline for the source deme is copied (subject to mutation) to the target deme, and an organism constructed from that germline is inserted into the target deme. The use of a digital germline has the side-effect of homogenizing the inhabitants of each deme, a technique that has been effective in evolutionary robotics [10]. Moreover, in an earlier study, we observed that using a digital germline was necessary to evolve organisms that cooperated to construct communication networks [16].

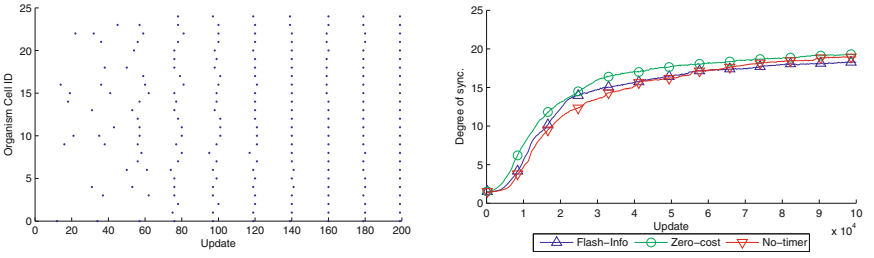
3 Experiments and Results

In this study we tested several different different combinations of instruction sets and environmental configurations for their ability to evolve synchronization behavior. Due to space limitations, here we only describe the configuration that was most effective; additional details can be found in an accompanying technical report [17]. We configured AVIDA with 400 5×5 toroidal demes, the default mutation rates (approximately 1 mutation per genome replication), and we employed a CompeteDemes period of 200 *updates*, where an update is the unit of virtual time in AVIDA corresponding to five virtual CPU cycles per organism. We initialized each deme to be in a state of desynchronization by starting from a single organism, and inserting exactly one new organism with a 50% probability each update. We configured the CompeteDemes framework to compete all 400 demes with each other every 200 updates, according to the following fitness function:

$$F = \begin{cases} 1 + U & \text{if } U < S \\ 1 + U + (flash_{max} - flash_{mean})^2 & \text{if } U \geq S \end{cases} \quad (1)$$

where F is the fitness of the deme, U is the number of unique organisms that issued a flash, S is the number of organisms in the deme (always 25 in this study), $flash_{max}$ is the maximum number of flashes during any single update, and $flash_{mean}$ is the mean number of flashes per update. Both $flash_{max}$, $flash_{mean}$, and U were calculated from the final 50 updates of each CompeteDemes period to allow organisms a time (the first 150 updates) during which they may issue flash instructions without adversely affecting the deme's fitness. This fitness function thus rewards demes whose constituent organisms each execute the flash instruction, and then further rewards demes for increasing the difference between the maximum and mean number of flashes per update. This definition of synchronization is similar to that in [7], where it is not required that all flashes occur at *exactly* the same point in time, but rather within a small window.

Figure 2(a) is a detail of the behavior of a single deme containing 25 copies of the *dominant* (that is, most common) evolved genome from the end of one of the best-performing of 30 AVIDA trials. Each point in this plot represents the execution of the flash instruction by a particular organism. Here we see runtime synchronization, where the 25 individuals within the deme have evolved to synchronize their flashes at an identical phase and frequency within 200 updates. Of note here is that evolution has solved *two* different problems: the period of successive flashes, and the behavior that brings them into synchronization.



(a) (b)

Fig. 2. Experimental results showing (a) detail of evolved synchronization behavior; organisms initially flash asynchronously, but gradually synchronize with each other, and (b) degree of synchronization measured as the difference between the maximum and mean number of flashes for three treatments.

To better understand the factors that influenced the evolution of synchronization, we conducted two additional treatments of the synchronization experiment. The *flash-info* treatment, which evolved the behavior seen in Figure 2(a), used a 20-cycle cost for execution of the flash instruction, and was allowed to use the flash-info instruction. The *zero-cost* treatment modifies the flash-info treatment by reducing the cost of executing the flash instruction to one cycle, making the flash instruction no more expensive to execute than any other instruction in the genome. Finally, the *no-timer* treatment modifies the flash-info treatment by replacing the flash-info instruction with flash-info-b, preventing organisms from accessing timing information related to the reception of flash messages.

Figure 2(b) plots the difference between the maximum and mean number of flashes, calculated as part of fitness, averaged over each deme in 30 AVIDA trials, for each of these three different treatments. Here we see that after 100,000 updates, the mean difference between maximum and mean flash counts approaches 20 in all three treatments. In other words, after 500 generations (100,000 updates / 200 updates per CompeteDemes period) 80% of the organisms within the average deme in each treatment synchronized with each other, and we see that this behavior does not depend on instruction cost or knowledge of the exact timing of message reception.

Finally, we examined the effect of a flash loss rate on the evolution of synchronization. Specifically, we varied loss rate from 0% to 80%, with each treatment using a unit-cost flash instruction. We define a flash "loss" as a flash message that is lost in sending, that is, it is not received by any of its neighbors. Figure 3 plots the number of coincident flashes versus time averaged over each deme, in 10 trials for each of the different flash loss rates from 0% to 80%. Here we see that higher loss rates inhibit the evolution of synchronization behavior, though we note that evolution was still able to discover solutions comparable to the 0% loss rate case at loss rates up to 20%.

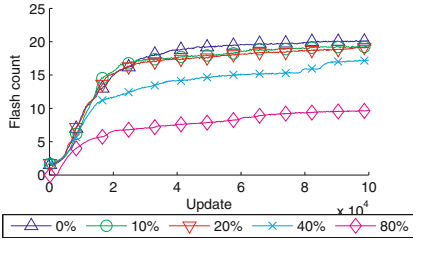
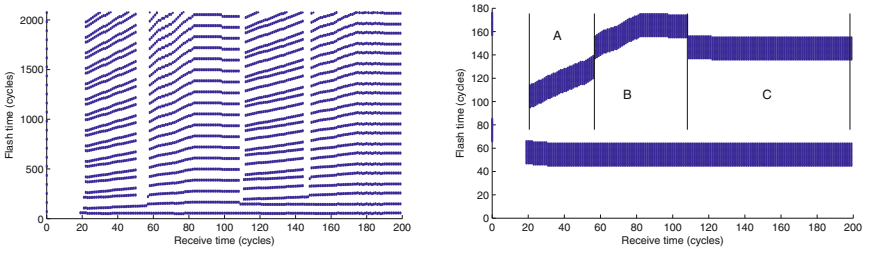


Fig. 3. Evolution of synchronization for different flash loss rates

4 Genome Analysis

Let is next analyze the genome responsible for the behavior shown in Figure 2(a), a dominant drawn from a single AVIDA trial. This treatment used a 20-cycle cost for the flash instruction, and included the flash-info instruction, which provided the organism with information about the last flash received. The AVIDA TestCPU enables the analysis of a single organism in an isolated environment. To analyze synchronization behavior, we extended the AVIDA TestCPU to support the artificial delivery of a flash message to the organism under test. We then traced the organism's execution flow that resulted from receiving a flash at different times, and monitored its response in terms of its own execution of the flash execution.

Figure 4(a) plots the response of the dominant to receiving a flash at various times during its life. At each point t along the horizontal axis, we initiated a new test of the dominant, and artificially sent a single flash message to the organism once it had executed t virtual CPU cycles. We then monitored the response of the organism from each test for 2000 cycles, and plotted a point for each cycle spent executing the flash instruction. For example, in Figure 4(a) at time 100, we see a series of horizontal bands every 110 cycles. This indicates that if the organism receives a single flash after it has executed 100 instructions, its resulting behavior will be to flash every 110 cycles. The gaps from time 1 to 19 and near times 50, 110, and 150 are artifacts of the organism executing the hard-reset instruction.



(a) (b)

Fig. 4. Genome analysis results showing (a) synchronization response of the dominant genome to receiving flash messages at different times, and (b) detailed synchronization response of this genome to receiving flash messages at different times. Regions A and B are frequency-increasing and frequency-decreasing responses, respectively, while region C is a steady-state response.

Figure 4(b) zooms in on the first 180 cycles of the organism's response to receiving flash messages at different times. Here we see evidence of the strategy employed for synchronization. First, the horizontal band extending from time 19 to 200 indicates that, regardless of flash reception, organisms will issue a flash that will complete by their 66th virtual CPU cycle. Region C in Figure 4(b), extending from time 108 to 200, shows that the organism has the same response to receiving a flash at any point in this region, indicating that the organism is not making any phase or frequency adjustments to its flash execution. Region A, on the other hand, shows an earlier execution of flash than region C, indicating that the reception of a message in this region causes a frequency-advance of the receiver. Finally, region B shows a slower response to flash

messages than region C, indicating a region of frequency-delay. This combination of steady-state and adaptive frequency operations is strikingly similar to the models of biological synchronization from Mirollo, Strogatz, and Ermentrout [5, 6].

Figure 5 depicts the instructions present in the dominant genome, and shows the primary instructions that are responsible for the behavior in Figure 4(b). Upon birth, the organism executes the first 64 instructions in order, unless reception of a flash has triggered a reset, as mentioned earlier. Otherwise, periodic execution of the flash instruction begins at cycle 65. When the organism receives a flash, however, its execution flow dramatically changes. Specifically, it uses a combination of the hard-reset, get-cycles, flash-info, and jmp-head instructions to modify the position of the virtual CPU's instruction pointer. Depending on where this instruction pointer is moved to, the next execution of flash will either be earlier, later, or unchanged relative to the organism's natural frequency (110 cycles, as shown by Figure 4(a)). The genome shown in Figure 5 is annotated with the regions corresponding to these different behaviors. The genome also makes extensive use of conditional logic, via the if-recvd-flash instruction, to retrieve the numeric position of the instruction pointer within the genome. Finally, we note the large number of instructions (75) within this genome that have no immediately discernible purpose (instructions not shown). These instructions are most likely neutral mutations, or mutations that do not adversely affect fitness, though they may have had an important role earlier in the evolutionary process.

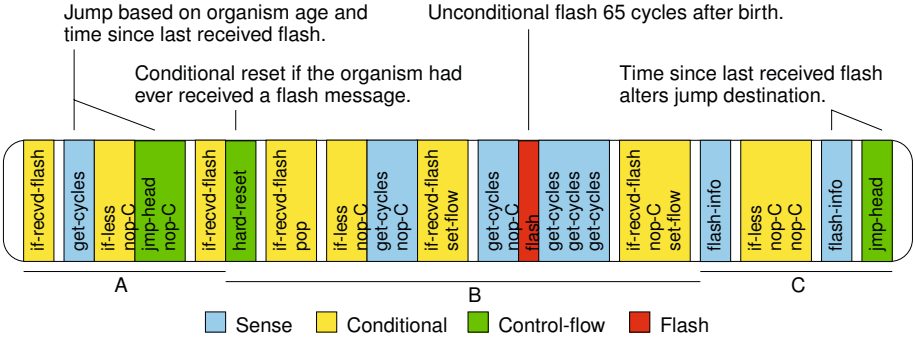


Fig. 5. Depiction of the dominant genome for synchronization. Labels (A, B, C) refer to destinations of the jmp-head instruction that correspond to frequency-advance, frequency-delay, and steady-state regions from Figure 4(b), respectively. Gaps in the genome show where instructions have been elided for clarity.

5 Conclusion

We have shown that digital evolution can evolve cooperative synchronization behaviors that are similar to behaviors observed in natural systems and the corresponding algorithms proposed recently for use in self-organizing computational systems. The evolved solutions utilized an adaptive frequency strategy similar to the Ermentrout model that altered their control flow based on a combination of sensed information and the organism's age. We have also shown that the evolution of synchronization is robust to

message loss. This result demonstrates that digital evolution shows promise as a means to produce relatively complex cooperative behaviors from very simple fitness functions.

Acknowledgments. This work was supported in part by NSF Grants CCF-0750787, CCF-0820220, and CNS-0751155; U.S. Army Grant W911NF-08-1-0495; and by a Quality Fund Grant from Michigan State University.

References

1. Southwick, E.E., Moritz, R.F.A.: Social synchronization of circadian rhythms of metabolism in honeybees (apis mellifera). Physiological Entomology 12(2), 209–212 (1987)
2. Backwell, P., Jennions, M., Passmore, N., Christy, J.: Synchronized courtship in fiddler crabs. Nature 391(6662), 31–32 (1998)
3. Holldobler, B., Wilson, E.O.: The Ants. Harvard University Press, Cambridge (1990)
4. Buck, J.: Synchronous Rhythmic Flashing of Fireflies, II. The Quarterly Review of Biology 63(3), 265–289 (1988)
5. Mirollo, R.E., Strogatz, S.H.: Synchronization of pulse-coupled biological oscillators. SIAM Journal on Applied Mathematics 50(6), 1645–1662 (1990)
6. Ermentrout, B.: An adaptive model for synchrony in the firefly pteroptyx malaccae. Journal of Mathematical Biology 29(6), 571–585 (1991)
7. Babaoglu, O., Binci, T., Jelasity, M., Montresor, A.: Firefly-inspired heartbeat synchronization in overlay networks. In: Proceedings of Self-Adaptive and Self-Organizing Systems (SASO), pp. 77–86 (2007)
8. Ofria, C., Wilke, C.O.: Avida: A software platform for research in computational evolutionary biology. Journal of Artificial Life 10, 191–229 (2004)
9. Nowak, M.A.: Five rules for the evolution of cooperation. Science 314(5805), 1560–1563 (2006)
10. Floreano, D., Mitri, S., Magnenat, S., Keller, L.: Evolutionary conditions for the emergence of communication in robots. Current Biology 17(6), 514–519 (2007)
11. Marocco, D., Nolfi, S.: Self-organization of communication in evolving robots. In: Proceedings of the Conference on Artificial Life (ALIFE), pp. 178–184 (2006)
12. Teuscher, C., Capcarrère, M.S.: On fireflies, cellular systems, and evolware. In: Tyrrell, A.M., Haddow, P.C., Torresen, J. (eds.) ICES 2003. LNCS, vol. 2606, pp. 1–12. Springer, Heidelberg (2003)
13. Ofria, C., Adami, C.: Evolution of genetic organization in digital organisms. In: Proceedings of DIMACS Workshop on Evolution as Computation (1999)
14. Knoester, D.B., McKinley, P.K., Beckmann, B.E., Ofria, C.: Directed evolution of communication and cooperation in digital organisms. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 384–394. Springer, Heidelberg (2007)
15. Beckmann, B.E., McKinley, P.K., Ofria, C.: Evolution of an adaptive sleep response in digital organisms. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 233–242. Springer, Heidelberg (2007)
16. Knoester, D.B., McKinley, P.K., Ofria, C.: Cooperative network construction using digital germlines. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO (2008)
17. Knoester, D.B., McKinley, P.K.: Evolving virtual fireflies. Technical Report MSU-CSE-08-9, Computer Science and Engineering, Michigan State University, East Lansing, Michigan (May 2008), http://www.cse.msu.edu/thinktank/firefly.pdf

Selection of Cooperative Partners in n-Player Games

Pedro Mariano[1], Luís Correia[2], and Carlos Grilo[2,3]

[1] Transverse Activity on Intelligent Robotics – IEETA – DETI, Universidade de Aveiro, Portugal
[2] LabMAg – Dep. de Informática, Faculdade de Ciências, Universidade de Lisboa, Portugal
[3] Dep. Eng. Informática, Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Leiria, Portugal

Abstract. We address the problem of finding the appropriate agents to interact with in n-player games. In our model an agent only requires knowledge about the payoff and identification of its partners. This information is used to update a probability distribution over candidate partners. As such, our model is applicable in any situation, be it a cooperative dilemma or a game where a Nash Equilibrium is equal to a Pareto Optimal profile.

1 Introduction

Reputation management [1,2], partner punishment [2], partner selection [3,4], network structure [5,6] are models put forward to explain or analyse the prevalence of cooperative agents in games where a dilemma is present. However, the extensibility of such approaches to any game is often not discussed, as the proposed solution only applies to a specific game [3,7,8].

As we use Game Theory to model the interactions between agents, our model makes direct use of the payoffs of a game in order to select the cooperative partners. The model consists of a probability vector maintained by each agent where each position represents the probability of selecting an agent as a partner to play a game. With this approach we are able to apply our model to any situation capable of being described as a game, with partner identification.

Since the agent has to find the best partner, the algorithm can be compared to a Cournot adjustment process [9] were players iteratively adjust their strategies to their partner responses. In this paper, an agent strategy remains constant but it adjusts its preferences towards more profitable or cooperative partners. Similar approaches to partner selection have been tackled in [3,4] but they focused on a specific game such as Prisoner's Dilemma (PD). Here we study the problem of partner selection in n-player games. In this case, the assessment of responsibility for the outcome is more difficult to make, due to increased uncertainty of having $n-1$ partners instead of a single one.

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 482–489, 2011.

2 Definitions

Game Theory is a tool to model interaction between agents. To this end, we consider that a population $\mathbb{P}$ of agents interacts accordingly to the rules of some n-player game $\mathcal{G}$. The game describes the strategies available to players and the payoffs they obtain as a function of the strategies used. The game has a n-dimensions strategy space $\mathbf{S} = \mathbb{S}_1 \times \mathbb{S}_2 \times \ldots \times \mathbb{S}_n$ where agents can draw a strategy $s \in \mathbb{S}_i$ to play a game. The vector $\mathbf{s} = (s_1, \ldots s_n)$ represents a strategy profile of the n players involved in the game. The game also has n payoff functions, $u_i : \mathbf{S} \rightarrow \mathbb{R}$, with $i \in \{1, 2, \ldots, n\}$. The payoff functions are bounded and belong to $\mathbb{R}$. Let $\underline{u}$ be the lowest payoff and $\overline{u}$ be the highest payoff in game $\mathcal{G}$.

We aim at reaching a position where cooperative agents only interact between themselves. As cooperative agents we define those that form a strategy profile that maximises the average payoff of the players. We define the payoff obtained by such Pareto Optimal profile as follows:

$$u_P = \max_{\mathbf{s}} \sum_i \frac{u_i(\mathbf{s})}{n}.$$

For example, in a Public Good Game a cooperative agent is one that contributes to the common good, and in the Common Pool Resource game a cooperative agent does not over exploit the resource (see [10] for a specification of these games).

3 Model Description

A population $\mathbb{P}$ of agents is represented by a directed simple graph where a vertex represents an agent α and $w_{\alpha,\beta}$ is the label of an edge from α to β representing the probability of agent α interacting with β:

$$w_{\alpha,\beta} \geq 0,$$
$$\sum_\beta w_{\alpha,\beta} = 1.$$

3.1 Update Policy

The edge weight update policy for agent α is a function defined as follows:

$$w_{\alpha,\beta}^{t+1} = \zeta(w_{\alpha,\beta}^t, u_\alpha)$$

where $w_{\alpha,\beta}^t$ is the edge weight before the game in which agent α participated, u_α is the payoff of the agent in the game. Index β varies through all neighbours of α.

The main focus of the work presented in this paper is the analysis of an update policy that meets the following two conditions:

Cooperative aggregation – Cooperative agents are mostly connected to each other. If α and the set of its cooperative neighbours B are part of a Pareto Optimal profile, then in the limit the sum of the probability of selecting only $\beta_C \in B$ should be 1:

$$\sum_{\beta_C \in B} \lim_{t\to\infty} w^t_{\alpha,\beta_C} = 1 \quad B = \{\beta : u_\alpha(\ldots, s_\beta, \ldots) = u_P\}. \tag{1}$$

Stability – The update policy must be robust in order to resist perturbations in the population and to be applicable to any n-player game. In the long run and in the absence of perturbations, weights must stabilise:

$$\lim_{t\to\infty} (w^{t+1}_{\alpha,\beta} - w^t_{\alpha,\beta}) = 0. \tag{2}$$

The edge weight update policy function is divided in two cases depending on whether an agent played the game with agent α or not.

Agent β Played the Game. A simple policy is to multiply the old weight by a factor that is inversely proportional to the distance between payoff u obtained by agent α and the Pareto Optimal payoff u_P if u is lower than u_P. If it is higher or equal, the edge weight remains the same. The rationale being there is no motive to decrease the probability of selecting the current partners. The definition is:

$$w^{t+1}_{\alpha,\beta} = \begin{cases} w^t_{\alpha,\beta} \dfrac{u - \underline{u}}{u_P - \underline{u}} & u < u_P \\ w^t_{\alpha,\beta} & u \geq u_P. \end{cases} \tag{3}$$

This rule by itself does not guarantee the condition in (1). Only combined with the rule for the case of agents that were not selected we achieve it. Regarding stability, this rule will keep weights unchanged if the payoff is greater or equal than u_P. Otherwise they will tend to zero as in the first case $w^t_{\alpha,\beta}$ is multiplied by a factor always less than 1. Either way, (2) is met.

Agent γ Did Not Play The Game. As we have just seen, the multiplicative factor used for agents that played the game implies that the weight of all agents that played will either stay the same or decrease. If they decrease, the difference must be distributed among the other edge weights. A simple solution is to distribute it equally:

$$w^{t+1}_{\alpha,\gamma} = w^t_{\alpha,\gamma} + \frac{s}{x} \tag{4}$$

where s is the sum of the differences of all link values, egressing node α, updated in the previous case,

$$s = \sum_{\beta} (w^{t+1}_{\alpha,\beta} - w^t_{\alpha,\beta})$$

and variable x is the number of neighbours of agent α that did not play.

This policy explores alternative partners if $u < u_P$, since the probability of selecting others in the next game round is increased.

Equation (4) combined with (3) are able to achieve the condition expressed by (1). If a cooperative agent selects an uncooperative, the corresponding weight will decrease towards zero. The difference is distributed among the weights of players that were not selected. However, the weight of a second uncooperative partner also increases, but not by much. If this second partner is selected, its weight is reduced and distributed among all the partners. The point is that, in the long run, weights of uncooperative agents decrease while weights of cooperative agents will absorb the distributed differences.

3.2 Credit Assignment Problem

In 2-player games an agent only has one partner. Therefore, the payoff it obtains in a game only depends on its strategy and the strategy of its partner (both fixed, but not necessarily identical). In these games, any player does not have doubts on the quality of its partner. In games with more than two players the situation is different.

In this type of games, there may be partners that the agent should favour instead of others. However, the payoff is not sufficient to establish a differentiated edge update policy. Recall that we have assumed that an agent only knows the payoff it obtains, besides the knowledge of who are its partners. We propose the following procedure:

In the first game, the agent selects $n-1$ partners randomly, plays the game, but does not apply the update policy. In the following games, the agent randomly replaces one partner and keeps the remaining $n-2$ partners. It observes the payoff obtained in game t and compares with the payoff obtained in game $t-1$. Let β_{t-1} be the partner that was selected in game $t-1$ and β_t the partner that replaced it. Since the agent has only changed one partner, it can compare the payoffs and see which should be favoured. The weight of the link to the agent which provided less payoff is updated using (3), weights of kept partners remain unchanged, and weights of remaining neighbours are updated using (4).

The drawback of this proposal is that it can only be applied by agent α in games where it selects its partners. The data obtained in games where it is selected by other agents (to play the game), cannot be used by this procedure. In order to use this data we need a more sophisticated agent, that is not analysed in this paper. However, in the next section, we show the usefulness of the simpler approach just described.

4 Experimental Analysis

The purpose of the experiments reported in this paper is to assess the convergence profile of cooperative agents selecting only their equals. To this end, simulations with different proportions and quantities of game strategies were performed.

4.1 The Game

The model presented in the previous section was tested with the Public Good Provision game [10]. The number of players ranged from 3 to 8. For results in 2-player games please refer to [11].

The Public Good Provision game is generally used to model situations where a group of persons has to contribute for a common good [12,13]. The more people contribute, the greater is the average payoff. However, contribution is costly, so shirking is the rational choice, provided there are at least some players who contribute to the good.

In the Public Good Provision game, an agent that contributes to the good, incurs in a cost c. The good is worth w. Let x be the proportion of agents that provides the good. The payoff of an agent that provides the good is $wx - c$ while agents that defect get wx. The game has a single iteration. The strategy used by agents is probabilistic and is defined by parameter p_p which is the probability to provide the good.

4.2 The Population

Regarding the agents, different strategies were used, which can be roughly classified in how cooperative they are:

S1 A cooperative strategy that always provides the good, ($p_p = 1.0$).
S2, S3 Two strategies that provide with probabilities 0.7 and 0.3, respectively.
S4 One uncooperative strategy that does not provide the good, ($p_p = 0.0$).

The number of strategies of each type varied in $\{8, 16, 32\}$. Total population varied in $\{32, 40, 48, \ldots, 128\}$. Moreover, we also performed simulations without strategies **S2** and **S3** thus having only deterministic strategies. This allows us to study convergence for different proportions and quantities of cooperative strategies. Initial edge weight was set to $1/(|\mathbb{P}| - 1)$ so that every player had the same chance of being selected.

Each simulation consisted of 1,000 rounds of games, except for one case where we ran 10,000 rounds, to test convergence. In each round all agents played at least one game, since the following steps were performed per round for every agent: select $n - 1$ partners proportionally to the edge weights, play the game, update the edge weights of the agent that selected partners.

4.3 Results

We have plotted the average probability of agents with strategy $p_p = 1.0$ to select agents from the four strategies. The plots also show the standard deviation. Unless mentioned, the plots were taken at the 1,000$^{\text{th}}$ round. Figure 1 shows simulations with only deterministic strategies. Figure 2 shows the results organised by number of cooperative strategies, while Fig. 3 shows the results organised by number of players in a game. Both latter figures refer to simulations with all four types of strategies. In all figures we plot an average taken over all possible population sizes and compositions. In Figs. 1 and 2 we also average results over all game configurations (3 to 8 players).

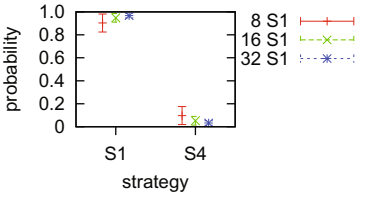
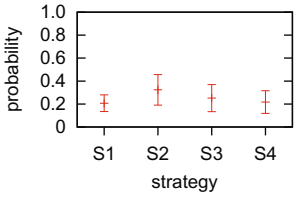
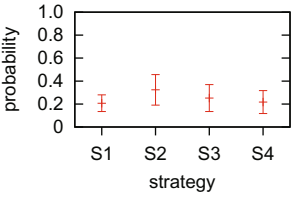


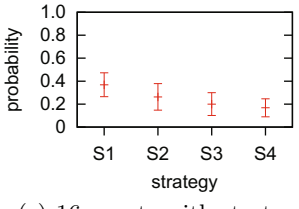
Fig. 1. Results from simulations with only deterministic strategies. Vertical axis represents the probability of strategy **S1** choosing a strategy in the horizontal axis. The results are averages of all possible combinations of strategy **S4** and number of players in the game.



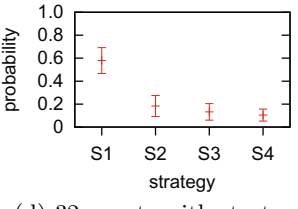
(a) 8 agents with strategy **S1**.



(b) 8 agents with strategy **S1**. Edge weights taken at the $10,000^{\text{th}}$ round.

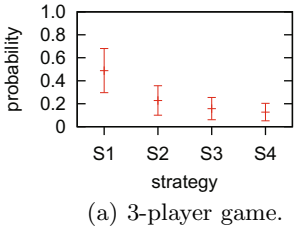


(c) 16 agents with strategy **S1**.

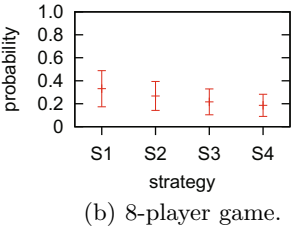


(d) 32 agents with strategy **S1**.

Fig. 2. Simulation results per number of cooperative strategies. Vertical axis represents the probability of strategy **S1** choosing a strategy in the horizontal axis. Results are averages of all combinations of parameters not fixed.



(a) 3-player game.



(b) 8-player game.

Fig. 3. Simulation results per number of players in game. Vertical axis represents the probability of strategy **S1** choosing a strategy in the horizontal axis. Results are averages of all combinations of parameters not fixed.

5 Discussion

In the absence of stochastic strategies, results show that cooperative agents almost always select their equals as partners independently of the conditions. From Fig. 1 we can observe that the average is high (for choosing **S1** strategies) while the standard deviation is small.

In the presence of stochastic strategies **S2** and **S3** results change substantially. The number of cooperators influences convergence: the more cooperators, the faster is convergence. We notice higher probability of cooperators selecting their equals for larger numbers of cooperators (compare Figs. 2(a), 2(c) and 2(d)).

One would expect that stochastic strategies would induce instability in convergence. Suppose we have partners (β_1, β_2) at round t. Both are stochastic and have provided in this round. If β_1 is replaced by a cooperative agent (always provides) and in the next round β_2 does not contribute, then this fact will be imputed to the cooperative agent. However, if we increase the number of rounds to 10,000, the plots obtained are nearly identical as can be seen by comparing Figs. 2(a) and 2(b), probabilities computed at the $1,000^{\text{th}}$ and $10{,}000^{\text{th}}$ rounds, respectively. This means that probabilities have converged to a stable situation by iteration 1,000.

The number of players in a game also influences convergence (see Figs. 3(a) and 3(b) that refer to 3 and 8 players, respectively). The higher is this number, the longer it takes for agents with **S1** strategy to only select themselves as partners of interaction. However, the time can be decreased if we increase the number of strategies **S1** in the population. Comparing Figs. 2(c) and 2(d), the latter refers to 32 agents with strategy **S1** and they select themselves as partners of interaction almost 60% of time. This value is higher than the former, 40%.

Notice that in spite of averaging a large number of simulations with different parameters (varying population size and composition, and also the number of players, for cases of Figs. 2 and 3), standard deviation is always small. We can infer that all these parameter values, except the number of cooperative agents, do not influence significantly the results.

Regarding uncooperative agents, they select cooperative agents as the partners of interaction when the number of **S1** strategies is high (16 or 32). When there are 8 agents with strategy **S1** we obtain a plot similar to the plot in Fig. 2(a).

6 Future Work

The model we have shown will be analysed in an evolutionary setting where strategies are replaced by some selection policy. Other network structures, besides the panmictic used in this paper, will also be considered.

Stochastic strategies impair convergence of deterministic cooperative strategies, when their number is low. Further analysis of their impact will be carried. The algorithm we have presented in Sect. 3.2 can be modified in order to distinguish deterministic from stochastic strategies. One avenue of researching this

is instead of changing one partner per round, keep the same partners when the payoff is equal or higher than u_P, and only change partners when the payoff is lower.

Acknowledgements

This work is partially supported by FCT/MCTES grant SFRH/BD/37650/2007.

References

1. Dellarocas, C.: Goodwill hunting: An economically efficient online feedback mechanism in environments with variable product quality. In: Falcone, R., Korba, L. (eds.) 5th Workshop on Deception, Fraud and Trust in Agent Societies, pp. 26–40 (2002)
2. Chalub, F., Santos, F.C., Pacheco, J.M.: The evolution of norms. Journal of Theoretical Biology 241(2), 233–240 (2006)
3. Biely, C., Dragosits, K., Thurner, S.: The prisoner's dilemma on co-evolving networks under perfect rationality. Physica D: Nonlinear Phenomena 228(1), 40–48 (2007)
4. Luthi, L., Giacobini, M., Tomassini, M.: Synchronous and asynchronous network evolution in a population of stubborn prisoners. In: CIG. IEEE, Los Alamitos (2005)
5. Santos, F.C., Santos, M.D., Pacheco, J.M.: Social diversity promotes the emergence of cooperation in public goods games. Nature 454, 213–216 (2008)
6. Hauert, C.: Spatial effects in social dilemmas. Journal of Theoretical Biology 240, 627–636 (2006)
7. Henrich, J., Boyd, R.: Why people punish defectors: Weak conformist transmission can stabilize costly enforcement of norms in cooperative dilemmas. Journal of Theoretical Biology 208, 79–89 (2001)
8. Barclay, P.: Reputational benefits for altruistic punishment. Evolution of Human Behaviour 27, 325–344 (2006)
9. Fisher, F.M.: The stability of the cournot oligopoly solution: The effects of speeds of adjustment and increasing marginal costs. The Review of Economic Studies 28(2), 125–135 (1961)
10. Gintis, H.: Game Theory Evolving - A problem-centered introduction to modeling strategic interaction. Princeton University Press, Princeton (2000)
11. Mariano, P., Correia, L., Grilo, C.: Analysis of a resource sharing game. To appear in 14th Portuguese Conference on Artificial Intelligence, EPIA 2009 (2009)
12. Blackwell, C., McKee, M.: Only for my own neighborhood? Preferences and voluntary provision of local and global public goods. Journal of Economic Behavior & Organization 52, 115–131 (2003)
13. van Dijk, F., Sonnemans, J., van Winden, F.: Social ties in a public good experiment. Journal of Public Economics 85, 275–299 (2002)

Evolving Social Behavior in Adverse Environments

Brian D. Connelly and Philip K. McKinley

Department of Computer Science and Engineering
Michigan State University
East Lansing, Michigan 48824, USA
{connel42,mckinley}@cse.msu.edu

Abstract. Cooperative behaviors are pervasive in the natural world. How organisms evolve stable cooperative strategies, specifically how selection can favor such costly behaviors, is a difficult problem for which several theories exist. In this work, we use digital evolution to explore the evolution of the production of a public resource that enables populations of organisms to survive in an adverse environment. Kin selection and limited dispersal are shown to promote cooperative acts, and evolved organisms stave off invasion by cheaters and survive in increasingly-adverse environments. Further, we observe how populations react to the disappearance and later re-emergence of adversity in the environment.

Keywords: digital evolution, cooperative behavior, kin selection, public resource.

1 Introduction

The evolution of stable cooperative acts, such as the sharing of food or resources [1], collaboration [2], and self-sacrifice [3], is difficult to explain, since the costs of these behaviors lower the fitness of the actors and make them exploitable by cheaters, who take advantage of the cooperative act, yet do not themselves contribute. One explanation is kin selection [4], whereby organisms maximize their *inclusive* fitness, which includes not only their individual fitness, but also the fitness of kin, who share either common ancestry or behavior. Because these recipients share genetic material, the reproductive successes of an organism's kin also benefit that organism. J.B.S. Haldane captured this idea in his oft-quoted statement, "I would lay down my life for two brothers or eight cousins." Organisms are able to target kin as recipients either through kin discrimination using expressed traits or implicitly if kin remain in close proximity.

Cooperative behaviors among microorganisms, in particular, have received considerable attention in recent years. Quorum sensing, or coordination mediated by chemicals deposited into the environment [5], is one such behavior. Other social acts, such as the production of iron-scavenging siderophores [6] or enzymes [7], have also been studied. Although researchers conducting work in these areas use sophisticated techniques, time often limits the number of generations which can be observed in wet lab experiments.

An alternate approach is to apply mathematical models. Game theory has been used to study the spread of behaviors in a population, most frequently with Prisoner's Dilemma or Snowdrift [8]. These models have provided insights into the dynamics of social behaviors, such as the effects of spatial structures [9], dynamic social ties [10],

G. Kampis, I. Karsai, and E. Szathmáry (Eds.): ECAL 2009, Part I, LNCS 5777, pp. 490–498, 2011.

and nonlinear benefits and resources [11]. One study using cellular automata [12] required agents to achieve a defined level of success in order to avoid being killed [13]. In these models, mutations altered the organism's level of investment in the public good. A similar approach is used by Evolution Strategies (ES) [14], which seek to optimize some behavior tied to the mutated parameter. These approaches generally do not include environmental conditions or features aside from the cooperative act, however.

Artificial life tools such as digital evolution [15] provide a means to address these difficulties. Given complete control over environmental conditions, researchers are able to accurately model complex systems. These tools also offer direct access to organisms' genomes, which greatly aids in understanding the behaviors observed and the conditions under which they are performed. For example, digital evolution has been used to examine group behaviors such as electing leaders, reaching consensus, population control, and foraging [16, 17]. Similar to the work described here, one study investigated kin discrimination in the altruistic suicide of colicinogenic bacteria [18], finding that cooperators were most successful against cheaters when their behaviors were more discriminatory.

In this paper, we describe a study using the Avida digital evolution platform to examine the evolution of cooperative behaviors in populations of organisms. These organisms were able to produce a resource that helped them to survive in adverse environments. When dispersal was limited, populations cooperated to prevent approximately 90% of their constituents from being killed. More well-mixed populations, while also successful, were not able to fare as well. Finally, we observe how populations reacted when the degree of adversity in the environment changed or when they were exposed to threats which they had not encountered in hundreds of generations.

2 The Avida Digital Evolution Platform

Avida is a software platform used to study the evolution of populations of self-replicating computer programs [15]. These computer programs, or "digital organisms," compete for space by completing *tasks* in a user-defined environment. As depicted in Figure 1, each digital organism exists independently within its own *cell* in the environment. An organism's behavior is defined by a circular list of instructions (its "genome"), which are executed sequentially on virtual hardware allocated to that organism. The virtual hardware comprises a CPU, three 32-bit registers, and two stacks. The CPU has four *heads*, which are used to control the execution flow of an organism's genome and aid in self-replication. Organisms control these components using a Turing-complete instruction set. During every *update*, each organism is allotted a number of CPU cycles for executing instructions in its genome.

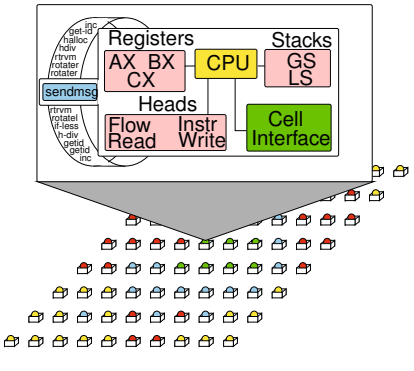


Fig. 1. The structure of a digital organism (top) in an Avida population (bottom) [19]

A population is seeded with an ancestral organism capable only of replication. All other behaviors exhibited by that organism's descendants must be evolved. As an organism replicates, it allocates additional space at the end of its genome into which it copies the instructions from its genome line by line. After this process has been completed, the genome is divided, and the new copy is placed into another cell in the environment, killing any organism at that location. During replication, mutations can cause the insertion, deletion, or changing of one or more instructions in the genome.

Organisms earn *merit* through the successful completion of *tasks*, which are defined by the user in terms of an organism's observable behaviors (i.e., its *phenotype*). Rewarded tasks may include performing a mathematical or logical operation [20], mitigating an attack [21], or cooperating to solve a distributed problem [17]. The number of CPU cycles an organism receives per update is directly proportional to its merit. This task mechanism creates competition in the population, as organisms with more merit are able to execute more quickly, and hence are likely to replicate more often and spread throughout the population. Tasks may involve the use of *resources*. By executing certain instructions, organisms are able to sense, consume, and produce resources. Resource levels in the environment can fluctuate over time through inflow, outflow, diffusion, decay, and consumption.

3 Experiments and Results

In this study, digital organisms evolved in environments in which a periodic event killed a portion of the population. An organism could avoid being killed if the amount of a resource in its cell was above a defined threshold. Although an organism could produce enough resource on its own to avoid being killed, that organism could also benefit from the resource production of its neighbors, due to the diffusion of the resource away from the cell in which it was created. Since the population exhibits limited dispersal, it is likely that neighbors are highly related, so kin selection [4] should enable this costly behavior to be maintained as long as it continues to provide a sufficient inclusive fitness benefit. However, since an organism can survive solely through the production of

Table 1. Logic tasks that could be completed by organisms. Upon completion, an organism's merit was multiplied by the reward listed. Organisms were rewarded once per task, but could continue to produce resource through multiple completions of OR NOT.

Task	Input	Output	Merit Bonus	Resource Produced
NOT	A	$\neg A$	2	0
NAND	A, B	$\neg(A \wedge B)$	2	0
AND	A, B	$A \wedge B$	4	0
OR	A, B	$A \vee B$	8	0
AND NOT	A, B	$A \wedge \neg B, \neg A \wedge B$	8	0
NOR	A, B	$\neg(A \vee B)$	16	0
XOR	A, B	$(A \wedge \neg B) \vee (\neg A \wedge B)$	16	0
EQU	A, B	$(A \wedge B) \vee (\neg A \wedge \neg B)$	32	0
OR NOT	A, B	$A \vee \neg B, \neg A \vee B$	0	1

resource by others, this environment also creates an opportunity for cheaters to exploit this public good.

Avida Configuration. For each experiment, multiple independent populations were evolved. Populations were seeded with a single ancestral organism that was placed into a cell in a 100x100-cell bounded grid environment; each interior cell had 8 neighboring cells. Two different replication methods were tested: limited dispersal, where offspring were placed in a cell neighboring their parent; and well-mixed, where offspring were placed in a random cell in the grid. During replication, an instruction mutated with probability 0.0075 during the copy phase, and an instruction was either added or removed with probability 0.05 as the organism divided. Each population started with a different random seed, so different evolutionary pathways were followed. These populations evolved for 50,000 updates, or approximately 7,000-10,000 generations.

Task Environment. Organisms evolved to complete one- and two-input logic tasks, listed in Table 1. When an organism executed an `IO` instruction, randomly-generated numbers were placed into its registers. The organism then had to perform the operation, place the result in the correct output register, and issue an additional `IO` instruction. If the output value matched the result of one of these tasks, the organism was rewarded.

Unlike other tasks, the completion of OR NOT did not yield any merit reward. Instead, it resulted in 1 unit of an extracellular resource being deposited into the environment at the organism's location. This resource both diffused and decayed at a rate of 1% per update and represented a beneficial product to the organism, akin to extracellular polymeric substance (EPS) [22], siderophores [6], or enzymes [23] in microorganisms. The amount of this resource present in a cell was used to determine whether or not an organism residing there was killed. Although this resource was beneficial to the organisms as described, it was not required for the completion of any other tasks.

Initial Experiments. Because each task could potentially serve as either a building block or a hindrance to completing another task, as in [20], we evolved populations in an environment without the periodic kill event in order to determine how many OR NOT tasks would be completed (and consequently how much resource would be produced) when this behavior was neither rewarded nor necessary for survival. These runs produced a mean per-cell resource level of 0.37 units. By using a significantly-higher threshold, we can infer that the success of organisms at staving off periodic killing was not simply a by-product of completing rewarded tasks.

Random Attacks. In this experiment, 121 cells (1.21% of the population) were randomly chosen at each update, and an organism living in those cells was killed if the level of resource in its cell was below 2 units. By the end of the run, organisms had evolved the production of the resource at a level that prevented approximately 66% of organisms selected from being killed. From this experiment alone, however, it is unclear whether the resource production was simply self-preserving or mutually-beneficial [24]. Since only 1.21% of organisms were subject to being killed at each update, the probability of an organism and all of its kin being killed was very small. Indeed, organisms in several runs preferred to play these odds and stopped producing resource, focusing entirely on completing the rewarded tasks.

Localized Attacks. In the next experiment, the kill event was localized, increasing the likelihood of one organism and its kin being killed by a single event. At each update, one cell was chosen at random, and any organism within a 5-cell radius (121 cells total) whose cell contained less than 2 units of resource was killed. The behaviors seen in these populations are plotted in Figure 2. At the end of these runs, an average of approximately 53% of organisms were cooperators (resource producers). Cheaters, which focused solely on the completion of one or more rewarded task but did not produce the resource, accounted for 23% of living organisms. Approximately 24% were unable to complete any tasks, most likely due to deleterious mutations.

We also tracked the level of resource in each cell as the runs progressed. One might predict this level to converge to the threshold amount of 2 units; however, modest stability was reached at approximately 9.5 units. This surplus can be viewed as cooperative: Cooperators produced enough to allow themselves to survive and to help their neighbors survive as well. This asymmetry follows the *Tragedy of the Commune* [25], where levels of investment in public goods may not be uniform. Figure 3(a) shows the distribution of this resource at the end of a typical run. Although only about half of the living organisms produced resource, it was sufficient to prevent 90% of organisms in the population from being killed as shown in Figure 3(b).

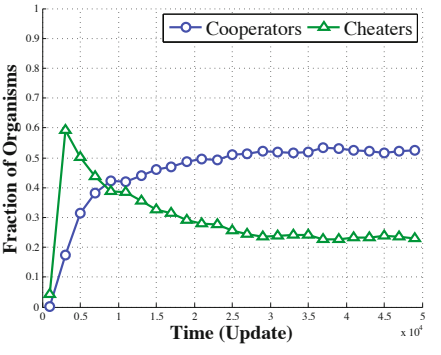
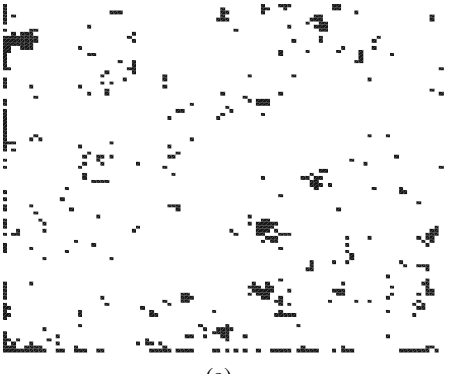
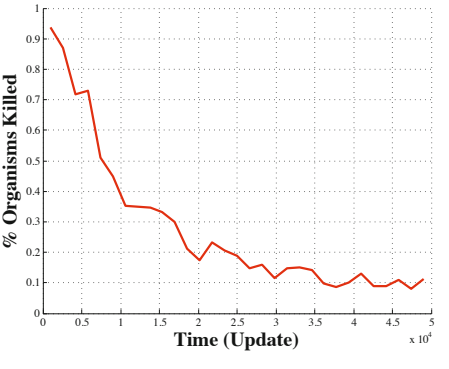


Fig. 2. Mean distribution of behaviors among 40 populations



(a)



(b)

Fig. 3. Resource distribution and organisms killed: (a) Distribution of resource levels in a typical environment. White indicates an above-threshold level, while black indicates a level below threshold. (b) Mean fraction of organisms killed within the target region.

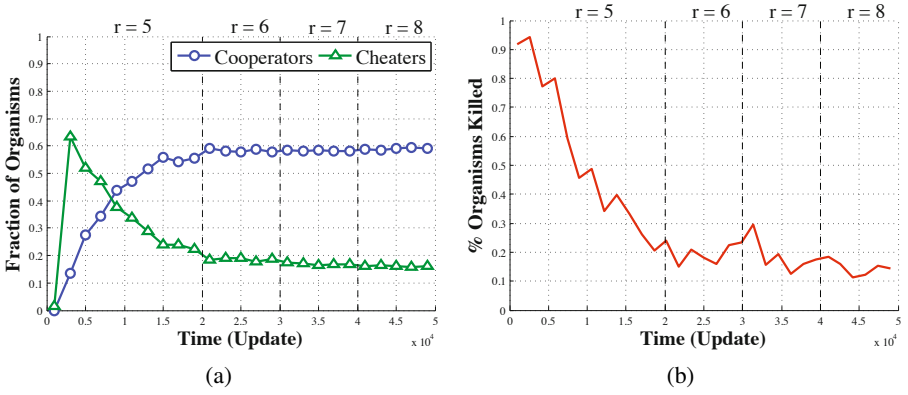


Fig. 4. Results for an increasingly-adverse environment: (a) Mean distribution of behaviors in 20 populations (b) Mean fraction of organisms within the target region killed during each kill event

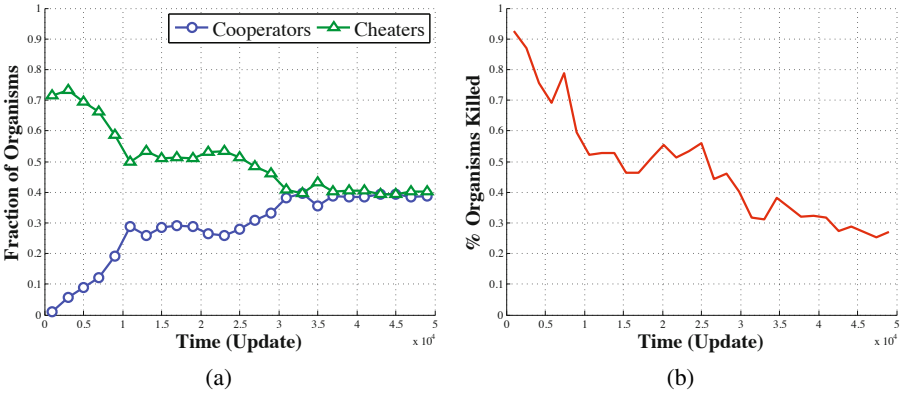
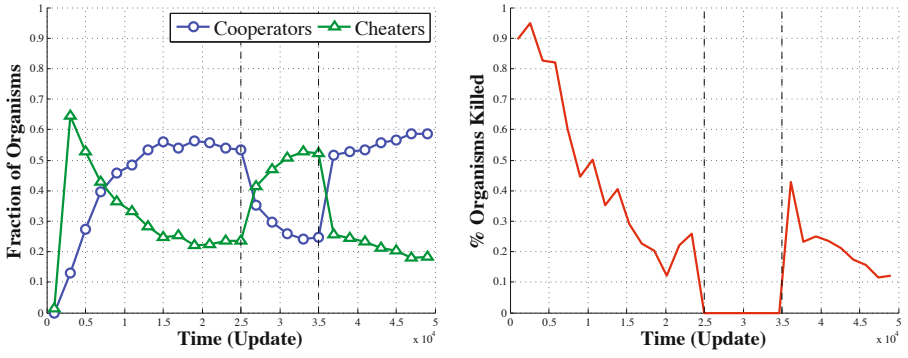


Fig. 5. Results in a well-mixed environment: (a) Mean distribution of cooperators and cheaters in 20 populations (b) Mean fraction of organisms killed within a 5-cell radius

Greater Adversity. To investigate how populations would evolve in more adverse environments, we repeated these experiments using kill radii up to 13 cells and a resource threshold of 3 units. These environments proved to be too adverse, and populations were not able to persist with kill radii above 5 cells. However, when the kill radius was expanded incrementally from 5 cells to 8, more than doubling the number of organisms at risk, the populations adapted and produced enough resource to survive these events. The distribution of resource producers and cheaters is shown in Figure 4(a). Here, organisms produced a mean resource level of 7.9 units per cell, enabling 85% of organisms to avoid being killed, as shown in Figure 4(b).

Well-Mixed Environment. The previous experiment demonstrated the benefits of kin selection to cooperative behaviors in populations with limited dispersal. Our next set of experiments examined how cooperation is affected when dispersal is increased. To



(a) (b)

Fig. 6. Results when the kill event was suspended between updates 25,000 and 35,000: (a) Mean distribution of behaviors (b) Mean fraction of organisms killed within the target region

achieve this, offspring were no longer placed into a neighboring cell, but instead into a random cell in the environment, greatly reducing the possibility of remaining near relatives. In these runs, cooperation emerged and persisted, resulting in approximately 40% of the population producing resource (Figure 5(a)). Interestingly, cheaters made up the same portion of the population. These results match those modeled with Snowdrift [8], where one's best strategy is to cooperate if their opponent cheats, and vice versa. These strategies enabled approximately 72% of organisms in kill regions to be spared, as shown in Figure 5(b).

Dynamic Conditions. To determine how populations would react to the absence and return of adversity, we evolved organisms in an environment where the kill event was suspended during updates 25,000 through 35,000. In these runs, offspring were once again placed into neighboring cells, and the kill event used a 5-cell radius with a 3-unit threshold. Figures 6(a) and 6(b) plot the strategies used by organisms and the fraction of organisms killed, respectively. During the respite, mean cellular resource levels fell below the kill threshold, leaving organisms vulnerable. Upon the return of the kill action, cooperative resource production re-evolved in 19 of the 20 populations.

4 Conclusions

This work has demonstrated that cooperative behaviors can evolve in populations exposed to adverse environments. Specifically, organisms evolved to complete tasks that did not provide direct bonuses as did other tasks, but instead produced a resource that helped prevent that organism and its neighbors from being killed. Kin selection was observed to provide incentive for developing such strategies when dispersal is limited. The increase in resource production by such populations indicates that this behavior is not simply selfish; rather, the behavior is mutually beneficial. We have also seen that

populations can quickly re-gain cooperative strategies in reaction to the return of adverse conditions after a calm period. Insights into cooperative behaviors offer potential applications as treatments for infectious disease, motivating future research in this area.

Acknowledgments

The authors gratefully acknowledge the contributions of Ben Beckmann, Art Covert, Tracy Teal and the MSU Devolab. This work was supported by NSF Grants CCF-0750787, CNS-0751155, CCF-0820220; by U.S. Army Grant W911NF-08-1-0495; and by a Quality Fund Grant from Michigan State University.

References

1. Kaplan, H., Hill, K.: Food sharing among ache foragers: Tests of explanatory hypotheses. Current Anthropology 26(2), 223 (1985)
2. Symonds, M.R., Elgar, M.A.: The evolution of pheromone diversity. Trends in Ecology & Evolution 23(4), 220–228 (2008)
3. Tofilski, A., et al.: Preemptive defensive self-sacrifice by ant workers. The American Naturalist 172(5), E239–E243 (2008)
4. Hamilton, W.D.: The genetical evolution of social behaviour. I,II. Journal of Theoretical Biology 7(1), 1–52 (1964)
5. Williams, P., et al.: Quorum sensing and the population-dependent control of virulence. Philosophical Transactions of the Royal Society B: Biological Sciences 355(1397), 667–680 (2000)
6. Griffin, A.S., West, S.A., Buckling, A.: Cooperation and competition in pathogenic bacteria. Nature 430, 1024–1027 (2004)
7. Burnham, J., Collart, S., Daft, M.: Myxococcal predation of the cyanobacterium Phormidium luridum in aqueous environments. Archives of Microbiology 137(3), 220–225 (1984)
8. Doebeli, M., Hauert, C.: Models of cooperation based on the Prisoner's Dilemma and the Snowdrift game. Ecol. Lett. 8(7), 748–766 (2005)
9. Nowak, M.A., May, R.M.: Evolutionary games and spatial chaos. Nature 359, 826–829 (1992)
10. Pestelacci, E., Tomassini, M.: Hawks and doves in an artificial dynamically structured society. In: Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems, pp. 466–473 (2008)
11. Gore, J., Youk, H., van Oudenaarden, A.: Snowdrift game dynamics and facultative cheating in yeast. Nature, 253–256 (2009)
12. Neumann, J.V.: Theory of self-reproducing automata. University of Illinois Press, US (1966)
13. Alonso, J., Fernández, A., Fort, H.: Prisoner's dilemma cellular automata revisited: Evolution of cooperation under environmental pressure. Journal of Statistical Mechanics: Theory and Experiment 2006(06), P06013 (2006)
14. Rechenberg, I.: Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution. Frommann-Holzboog (1973)
15. Ofria, C., Wilke, C.: Avida: A software platform for research in computational evolutionary biology. Artificial Life 10(2), 191–229 (2004)
16. McKinley, P.K., Cheng, B., Ofria, C., Knoester, D., Beckmann, B.E., Goldsby, H.: Harnessing digital evolution. IEEE Computer 41(1), 54–63 (2008)

17. Connelly, B.D., McKinley, P.K., Beckmann, B.E.: Evolving cooperative pheromone usage in digital organisms. In: Proceedings of IEEE Symposium on Artificial Life, pp. 184–191 (2009)
18. Goings, S., Clune, J., Ofria, C., Pennock, R.T.: Kin-selection: The rise and fall of kin-cheaters. In: Proceedings of the Ninth International Conference on Artificial Life, pp. 303-308 (2004)
19. Knoester, D.B., McKinley, P.K.: Cooperative network construction using digital germlines. In: Proceedings of the Genetic and Evolutionary Computation Conference, Atlanta, Georgia (2008)
20. Lenski, R.E., Ofria, C., Pennock, R.T., Adami, C.: The evolutionary origin of complex features. Nature 423, 139–144 (2003)
21. Beckmann, B.E., McKinley, P.K.: Evolution of adaptive population control in multi-agent systems. In: Proceedings of the Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems, Venice, Italy, pp. 181–190 (2008)
22. Rainey, P.B., Rainey, K.: Evolution of cooperation and conflict in experimental bacterial populations. Nature 425, 72–74 (2003)
23. Carlson, M., Botstein, D.: Two differentially regulated mRNAs with different 5' ends encode secreted and intracellular forms of yeast invertase. Cell 28(1), 145–154 (1982)
24. West, S.A., Griffin, A.S., Gardner, A., Diggle, S.P.: Social evolution theory for microorganisms. Nature Reviews: Microbiology 4, 597–607 (2006)
25. Doebeli, M., Hauert, C., Killingback, T.: The evolutionary origin of cooperators and defectors. Science 306, 859–862 (2004)

Author Index