

**Adrian-Horia Dediu  
Carlos Martín-Vide  
Bianca Truthe  
Miguel A. Vega-Rodríguez (Eds.)**

**LNCS 8273**

# **Theory and Practice of Natural Computing**

**Second International Conference, TPNC 2013  
Cáceres, Spain, December 2013  
Proceedings**

 **Springer**

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

## Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Alfred Kobsa

*University of California, Irvine, CA, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*TU Dortmund University, Germany*

Madhu Sudan

*Microsoft Research, Cambridge, MA, USA*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Gerhard Weikum

*Max Planck Institute for Informatics, Saarbruecken, Germany*

Adrian-Horia Dediu Carlos Martín-Vide  
Bianca Truthe Miguel A. Vega-Rodríguez (Eds.)

# Theory and Practice of Natural Computing

Second International Conference, TPNC 2013  
Cáceres, Spain, December 3-5, 2013  
Proceedings



Springer

## Volume Editors

Adrian-Horia Dediu

Rovira i Virgili University, Research Group on Mathematical Linguistics  
Avinguda Catalunya, 35, 43002 Tarragona, Spain  
E-mail: adrian.dediu@urv.cat

Carlos Martín-Vide

Rovira i Virgili University, Research Group on Mathematical Linguistics  
Avinguda Catalunya, 35, 43002 Tarragona, Spain  
E-mail: carlos.martin@urv.cat

Bianca Truthe

Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik  
Institut für Wissens- und Sprachverarbeitung  
Universitätsplatz 2, 39106 Magdeburg, Germany  
E-mail: truthe@iws.cs.uni-magdeburg.de

Miguel A. Vega-Rodríguez

University of Extremadura, Polytechnic School  
Department of Technologies of Computers & Communications (TC2)  
Campus Universitario s/n, 10003 Cáceres, Spain  
E-mail: mavega@unex.es

ISSN 0302-9743

e-ISSN 1611-3349

ISBN 978-3-642-45007-5

e-ISBN 978-3-642-45008-2

DOI 10.1007/978-3-642-45008-2

Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013952733

CR Subject Classification (1998): F.1, I.2, C.2, F.2, H.4, J.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

© Springer-Verlag Berlin Heidelberg 2013

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

*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

This volume contains the papers presented at the Second International Conference on the Theory and Practice of Natural Computing (TPNC 2013) held in Cáceres, Spain, during December 3–5, 2013.

The scope of TPNC is rather broad, containing topics of either theoretical, experimental or applied interest. The topics include but are not limited to:

## Nature-inspired models of computation

- amorphous computing
- cellular automata
- chaos and dynamical systems based computing
- evolutionary computing
- membrane computing
- neural computing
- optical computing
- swarm intelligence

## Synthesizing nature by means of computation

- artificial chemistry
- artificial immune systems
- artificial life

## Nature-inspired materials

- computing with DNA
- nanocomputing
- physarum computing
- quantum computing and quantum information
- reaction-diffusion computing

## Information processing in nature

- developmental systems
- fractal geometry
- gene assembly in unicellular organisms
- rough/fuzzy computing in nature
- synthetic biology
- systems biology

Applications of natural computing to algorithms, bioinformatics, control, cryptography, design, economics, graphics, hardware, learning, logistics, optimization, pattern recognition, programming, robotics, telecommunications, etc.

TPNC 2013 received 47 submissions. Each one was reviewed by three Program Committee members and there were also several external referees. After a thorough and vivid discussion phase, the committee decided to accept 18 papers

(which represents an acceptance rate of 38.30%). The conference program also included one invited talk and one invited tutorial.

Part of the success in the management of the submissions and reviews is due to the excellent facilities provided by the EasyChair conference management system.

We would like to thank all invited speakers and authors for their contributions, the Program Committee and the external reviewers for their cooperation, Diputación de Cáceres for the excellent facilities put at our disposal, and Springer for its very professional publishing work.

September 2013

Adrian-Horia Dediu  
Carlos Martín-Vide  
Bianca Truthe  
Miguel A. Vega-Rodríguez

# Organization

## Program Committee

Selim G. Akl	Queen's University at Kingston, Canada
Thomas Bäck	Leiden University, The Netherlands
Peter J. Bentley	University College London, UK
Hans-Georg Beyer	University of Applied Sciences at Dornbirn, Austria
Mauro Birattari	Université Libre de Bruxelles, Belgium
Jinde Cao	Southeast University, China
Vladimir Cherkassky	University of Minnesota at Minneapolis, USA
Sung-Bae Cho	Yonsei University, South Korea
John A. Clark	University of York, UK
Carlos A. Coello Coello	National Polytechnic Institute, Mexico
David W. Corne	Heriot-Watt University, UK
Peter Dayan	University College London, UK
Bernard De Baets	Ghent University, Belgium
Enrique Herrera-Viedma	University of Granada, Spain
Yaochu Jin	University of Surrey, UK
Nikola Kasabov	Auckland University of Technology, New Zealand
Vladik Kreinovich	University of Texas at El Paso, USA
Kwong-Sak Leung	Chinese University of Hong Kong, China
Xiaohui Liu	Brunel University, UK
Manuel Lozano	University of Granada, Spain
Carlos Martín-Vide (Chair)	Rovira i Virgili University, Spain
Julian F. Miller	University of York, UK
Frank Neumann	University of Adelaide, Australia
Leandro Nunes de Castro	Mackenzie Presbyterian University, Brazil
Nikhil R. Pal	Indian Statistical Institute, India
Günther Palm	University of Ulm, Germany
José Carlos Príncipe	University of Florida at Gainesville, USA
Helge Ritter	Bielefeld University, Germany
Conor Ryan	University of Limerick, Ireland
Hava Siegelmann	University of Massachusetts at Amherst, USA
Moshe Sipper	Ben Gurion University, Israel
Thomas Stützle	Université Libre de Bruxelles, Belgium

## VIII Organization

Ponnuthurai N. Suganthan	Nanyang Technological University, Singapore
Johan Suykens	Catholic University of Leuven, Belgium
Kay Chen Tan	National University of Singapore, Singapore
Dacheng Tao	University of Technology Sydney, Australia
Jon Timmis	University of York, UK
Marco Tomassini	University of Lausanne, Switzerland
Michael N. Vrahatis	University of Patras, Greece
Michael D. Vose	University of Tennessee at Knoxville, USA
Harald Weinfurter	University of Munich, Germany
Rolf Würtz	Ruhr University, Germany
Jun Zhang	Sun Yat-sen University, China

## External Reviewers

Bhalla, Navneet	Liu, Qingshan
Brambilla, Manuele	Nallaperuma, Samadhi
Ferrari, Daniel	Pasti, Rodrigo
Gee, Sen Bong	Piscopo, Carlotta
Hong, Zhibin	Szabo, Alexandre
Hu, Jianqiang	

## Organizing Committee

Adrian-Horia Dediu, Tarragona  
Carlos Martín-Vide, Tarragona (Co-chair)  
Bianca Truthe, Magdeburg  
Miguel A. Vega-Rodríguez, Cáceres (Co-chair)  
Florentina-Lilica Voicu, Tarragona

## Local Committee

Víctor Berrocal-Plaza  
José M. Chaves-González  
Juan A. Gómez-Pulido  
David L. González-Álvarez  
José M. Granado-Criado  
Alejandro Hidalgo-Paniagua  
José M. Lanza-Gutiérrez  
Álvaro Rubio-Largo  
Sergio Santander-Jiménez  
Miguel A. Vega-Rodríguez (Chair)



# Table of Contents

## Invited Talks

Boosting Interactive Evolution Using Human Computation Markets . . . . .	1
<i>Joel Lehman and Risto Miikkulainen</i>	

## Regular Papers

A New Version of the Multiobjective Artificial Bee Colony Algorithm for Optimizing the Location Areas Planning in a Realistic Network . . . . .	19
<i>Victor Berrocal-Plaza, Miguel A. Vega-Rodríguez, and Juan M. Sánchez-Pérez</i>	

Yield Optimization Strategies for (DNA) Staged Tile Assembly Systems . . . . .	31
<i>Eugen Czeizler and Pekka Orponen</i>	

Online Gait Learning for Modular Robots with Arbitrary Shapes and Sizes . . . . .	45
<i>Massimiliano D'Angelo, Berend Weel, and A.E. Eiben</i>	

Approach for Recognizing Allophonic Sounds of the Classical Arabic Based on Quran Recitations . . . . .	57
<i>Yahya O. Mohamed Elhadj, Mansour Alghamdi, and Mohamed Alkanhal</i>	

Reliable Self-assembly by Self-triggered Activation of Enveloped DNA Tiles . . . . .	68
<i>Vinay Kumar Gautam, Pauline C. Haddow, and Martin Kuiper</i>	

Learning from Uncertain Data Using Possibilistic Artificial Immune Recognition Systems . . . . .	80
<i>Rim Hentech, Ilyes Jenhani, and Zied Elouedi</i>	

An Efficient Biomechanical Cell Model to Simulate Large Multi-cellular Tissue Morphogenesis: Application to Cell Sorting Simulation on GPU . . . . .	96
<i>Anne Jeannin-Girardon, Pascal Ballet, and Vincent Rodin</i>	

Computational Modelling of the Interruptional Activities between Transposable Elements . . . . .	108
<i>Lingling Jin and Ian McQuillan</i>	

Probabilistic Analysis of Long-Term Swarm Performance under Spatial Interferences . . . . .	121
<i>Yara Khaluf, Mauro Birattari, and Franz Rammig</i>	
Improving MLP Neural Network Performance by Noise Reduction . . . . .	133
<i>Mirostaw Kordos and Andrzej Rusiecki</i>	
A Trajectory Algorithm to Solve the Relay Node Placement Problem in Wireless Sensor Networks . . . . .	145
<i>Jose M. Lanza-Gutiérrez, Juan A. Gomez-Pulido, and Miguel A. Vega-Rodríguez</i>	
Using Dynamic, Full Cache Locking and Genetic Algorithms for Cache Size Minimization in Multitasking, Preemptive, Real-Time Systems . . . . .	157
<i>Antonio Martí Campoy, Francisco Rodríguez-Ballester, and Rafael Ors Carot</i>	
General Quantum Encryption Scheme Based on Quantum Memory . . . . .	169
<i>Marius Nagy and Naya Nagy</i>	
Quantum Secret Communication without an Encryption Key . . . . .	181
<i>Marius Nagy and Naya Nagy</i>	
Securely Computing the Three-Input Majority Function with Eight Cards . . . . .	193
<i>Takuya Nishida, Takaaki Mizuki, and Hideaki Sone</i>	
An Analysis of a Selecto-Lamarckian Model of Multimemetic Algorithms with Dynamic Self-organized Topology . . . . .	205
<i>Rafael Nogueras, Carlos Cotta, Carlos M. Fernandes, Juan Luis Jiménez Laredo, Juan Julián Merelo, and Agostinho C. Rosa</i>	
Parallel Multi-Objective Genetic Algorithm: GPU Accelerated Asynchronously Distributed NSGA II . . . . .	217
<i>Oliver Rice, Robert E. Smith, and Rickard Nyman</i>	
Evolutionary Scheduling for Mobile Content Pre-fetching . . . . .	228
<i>Omar K. Shoukry and Magda B. Fayek</i>	
<b>Author Index . . . . .</b>	<b>241</b>

# Boosting Interactive Evolution Using Human Computation Markets

Joel Lehman and Risto Miikkulainen

Department of Computer Science, The University of Texas at Austin, USA  
{joel,risto}@cs.utexas.edu

**Abstract.** Interactive evolution, i.e. leveraging human input for selection in an evolutionary algorithm, is effective when an appropriate fitness function is hard to quantify yet solution quality is easily recognizable by humans. However, single-user applications of interactive evolution are limited by *user fatigue*: Humans become bored with monotonous evaluations. This paper explores the potential for bypassing such fatigue by directly purchasing human input from human computation markets. Experiments evolving aesthetic images show that purchased human input can be leveraged more economically when evolution is first seeded by optimizing a purely-computational aesthetic measure. Further experiments in the same domain validate a system feature, demonstrating how human computation can help guide interactive evolution system design. Finally, experiments in an image composition domain show the approach's potential to make interactive evolution scalable even in tasks that are not inherently enjoyable. The conclusion is that human computation markets make it possible to apply a powerful form of selection pressure mechanically in evolutionary algorithms.

## 1 Introduction

A critical component of any evolutionary computation (EC) experiment is selection, i.e. how the parents of the next generation are chosen from the current population. In particular, the success of a particular EA in a given domain often depends upon choosing an appropriate *fitness function* to guide search. That is, for an EA to produce a solution, the fitness function that is optimized must induce a sufficiently smooth gradient of increasing fitness that leads from the random individuals in the initial population to a solution. However, intuitive choices for fitness functions may often fail to identify the intermediate steps that lead to the solution [10,4], and some concepts intuitive to humans remain difficult to quantify algorithmically [17,19].

For example, creating an algorithmic characterization of aesthetic appeal to automate evolving aesthetic artifacts is a compelling [11,3,13] yet unfulfilled endeavor [17,13]. In such cases, one way to bypass this lack of an algorithmic measure is through interactive evolutionary computation (IEC; [19]), wherein humans act as a fitness function, actively selecting which solutions to evolve further. The insight is that humans may be able to evaluate a characteristic even when it cannot be mechanically recognized or rigorously defined.

However, a significant problem in IEC is user fatigue: A single user can only perform so many evaluations before becoming tired or bored [19]. A recent solution to this problem is to create collaborative IEC websites whereby without financial incentive users cooperate to evolve complex artifacts they could not have evolved alone [17,2,12]. The idea is that although a single user may become fatigued, if that user *publishes* their work on the website, other users can choose to further *extend* that published work. Over time, the artifacts that are published can accumulate and form a branching phylogeny of diverse and interesting content [17].

This approach is economical and promising when task domains are inherently enjoyable, e.g. creative domains like open-ended image, shape, or music evolution [17,2,12]. However, when attempting to apply the approach to arbitrary domains there are two significant limitations: (1) sustained evolution for many generations depends upon the task domain being engaging enough to continually draw a sufficient volume of volunteer users, and (2) implementing the idea requires creating the non-trivial system architecture that composes a collaborative evolution website, e.g. architecture that supports creating and handling user accounts, facilitating discovering and rating artifacts, and evolving and publishing artifacts.

An interesting potential solution to these problems is provided by human computation markets (HCMs). In these markets it is possible to pay for human input in arbitrary tasks and thereby keep humans motivated even when the task is not particularly rewarding itself. This paper explores whether HCMs can be effectively used for this purpose, through an approach called HCM+IEC that uses HCMs to perform selection in an interactive evolutionary algorithm.

The paper focuses on three ideas: First, even if the domain to be used with IEC is itself enjoyable and engaging (e.g. evolving aesthetic images), IEC websites face the bootstrapping problem common to all user-generated content sites. That is, at such a site's launch, when attracting users is most important, the site is *least* engaging due to lack of content. Thus the first contribution of this paper is to suggest that markets for human computation can help overcome this bootstrap problem: Initially users can be paid to generate content. For this reason, experiments with such an aim apply IEC+HCM in an image evolution domain representative of those often explored by collaborative IEC websites. The results show that human computation can be more efficiently leveraged if a computational aesthetic measure [11] first algorithmically generates an interesting diversity of images upon which humans can further elaborate.

Second, when designing an IEC website or a single-user IEC system, often many design decisions about the underlying algorithm must be made that will significantly impact the quality of the system's output. Problematically however, such important decisions often are guided only by the preferences and intuitions of the system designers. The second contribution of this paper is thus to suggest that IEC+HCM can be applied to conduct controlled experiments that measure the impact of a design decision on the quality of an IEC system's products. Experiments in the same image evolution domain show that removing a significant

feature results in measurably less aesthetically pleasing pictures, thereby demonstrating the potential for IEC+HCM to facilitate principled IEC system design.

Third, there are EC problems that could benefit from large-scale human selection but for which a collaborative IEC website will not be a feasible solution. That is, most current IEC websites rely on self-directed users to produce content, and such content is produced irregularly and only to the extent that volunteer users *enjoy* evaluating artifacts in the domain. Thus the third contribution of this paper is to demonstrate how IEC+HCM can be used instead of a collaborative IEC website in one such condition, i.e. when the task domain is not enjoyable.

The conclusion is that HCMs offer a mechanism for converting money into a powerful form of selection pressure that may prove a valuable tool for interactive evolution.

## 2 Background

In this section, the foundational technologies applied in the experiments in this paper, including interactive evolution, human computation, and heuristics for evolving impressive artifacts, are reviewed.

### 2.1 Interactive Evolution

Applying human judgment to perform selection in an evolutionary algorithm is called interactive evolutionary computation (IEC; [19]) and is motivated by the difficulty in quantifying intuitive concepts that are readily recognized by humans (e.g. aesthetic appeal), and also by the impressive examples of human-directed breeding (e.g. the wide variety of domesticated dogs or the increased potency of human-bred agricultural crops). While IEC has been explored in the context of single-user applications [19], collaborative websites [17,2], and online video games [6,16], it has only been superficially explored in the context of HCMs [1], i.e. websites that facilitate paying human users to complete small tasks that cannot be easily algorithmically automated.

Note that although both combine human intuition with evolutionary algorithms, IEC is distinct from Human-Based Genetic Algorithms (HBGAs; [9]). In HBGAs, humans perform not only selection (as in IEC), but implement *all* genetic operators and additionally serve as the substrate for genetic representation. That is, a human participating in a HBGA might recombine two existing pictures by *drawing* a new image that combines high-level features of both, instead of *breeding* together pictures generated by an underlying computational genetic system as in IEC. While HBGAs have been previously combined with HCMs [24,23], they impose the requirement that humans be able to construct and manipulate the artifacts being evolved. In other words, often recognizing a promising artifact is easier than creating one. Thus, a potential advantage of IEC is that one can breed complex artifacts (e.g. neural networks or complex

pictures) without understanding their construction. In this way, IEC can enable humans without expert knowledge to aid in solving difficult computational problems.

Supporting this idea, previous studies with IEC have demonstrated its promise for evolving complex structures, e.g. significantly increasing efficiency when evolving artificial neural networks (ANNs) that control mobile robots [5,22]. Also, in situations where large numbers of evaluations are impractical, it has been shown that IEC can make problems more tractable [5].

A representative example of a scalable IEC approach is given by the Picbreeder website [17], which encourages indirect user collaboration to evolve aesthetic images. On the site, users can discover, rate, and extend (through further interactive evolution) previously evolved images that are represented by compositional pattern producing networks (CPPNs; [18]). Note that CPPNs are feed-forward ANNs with an extended set of activation functions chosen for the regularities they induce [18]. The success of Picbreeder in evolving a wide variety of interesting and complex images likely stems from combining together an open-ended genetic encoding, an open-ended domain, and a powerful form of selection pressure (i.e. human judgment). Thus, similarly designed websites may be one path to large-scale IEC and compelling evolved artifacts.

However, evolution in such websites is typically undirected (i.e. driven by users' whims on what to create) and public (i.e. driven by users' ability to discover and elaborate upon existing content); for commercial IEC applications the ability to more directly guide the evolutionary process may be important and additionally it may be necessary for evolved content to be kept private (i.e. not stored such that all content is publicly accessible).

These limitations motivate exploring new approaches for large-scale IEC. A promising resource that can be leveraged for such purposes is human computation, which is reviewed next.

## 2.2 Human Computation

While the range of tasks solvable by computers continues to expand, there remain tasks that are challenging to solve computationally but are trivial for humans to solve. Examples of such tasks are recognizing written text [21], identifying objects in images [20], or evaluating aesthetic appeal [13]. This asymmetry motivates leveraging *human computation* [21] to automatically integrate human insight into algorithmic processes. Such human computation can often be made more scalable by employing *crowdsourcing* [8,15], whereby many small contributions from a diffuse group of people (often online) are aggregated.

For example, while CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) separate humans from machines by generating tasks that are easily solvable by humans but difficult for machines, the widely-deployed reCAPTCHA system [21] acts as a CAPTCHA while at the same time leveraging human computation to transcribe words from old books. That is, part of each word identification task posed by reCAPTCHA to its users is not machine generated, but is an image of a phrase that algorithmic

character recognition struggled to classify automatically. Similarly, “games with a purpose” (GWAP; [20]) are designed such that human enjoyment results from deriving and verifying solutions to problems that are not yet solvable computationally; in this way, game players cooperate to create tagged data sets as a byproduct of an enjoyable experience [20].

ReCAPTCHA and GWAPs show that sometimes users can be enticed to generate useful computation without economic incentive. However, it is unclear how to transform an arbitrary human computation task into an enjoyable or necessary process such that the task’s solution is a byproduct. Additionally, rather than wrap a task in a cleverly designed game and attempt to attract volunteers to play it, it may be simpler or cheaper sometimes to simply *pay* a human to perform the desired task through a HCM. The most well-known such marketplace is the Amazon Mechanical Turk (AMT; [7]), which is the system used in the experiments in this paper.

AMT exposes an interface to programmers that allows them to upload human intelligence tasks (HITs) which specify a desired task, an interface for humans to perform it, and the monetary reward for successfully completing the task. Once a human completes the HIT, the results can be queried and approved so that the human user can be paid. In this way, markets for human computation like AMT allow seamless integration of algorithms with arbitrary human input through economic exchange.

The next section reviews an algorithmic aesthetic measure that seeds evolution in some experiments in this paper.

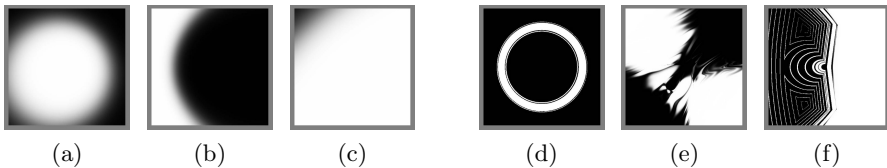
### 2.3 Evolving Impressive Artifacts

One dimension of what humans appreciate as impressive is the perceived amount of design effort necessary to create an artifact. In other words, it is *easy* to recognize how *difficult* an impressive artifact was to create [11]. For example, intuitively it is easier to recognize a good novel than it is to write one, and it is easier to perceive a back-flip than it is to perform one. Supporting such an idea, artifacts often appear less impressive if they require significant effort to create but such effort is not readily apparent, e.g. a painting of something trivial that is entirely indistinguishable from a photograph. Conversely, when tasks are trivial, the disparity in effort between recognition and creation is much less; for example, reading a novel composed of random words may take more effort than actually writing one. Put another way, it is easy to verify the difficulty in creating an impressive artifact. Interestingly, characterizing impressiveness this way parallels the idea of NP-completeness: Solutions to NP-complete problems are easily verified but difficult to derive.

Although a philosophical description of impressiveness may be thought-provoking, its application to computational experiments is limited unless it can be quantified. Lehman and Stanley [11] introduced two heuristics for estimating algorithmically the design effort necessary to recreate an artifact: rarity and recreation effort. That is, because the ability to perform a back-flip is rare, such rarity may indicate that back-flips are an impressive act. Similarly, the relative

rarity of a noticeable property or combination of such properties in a wider space may hint that they are impressive; for example, an image with a symmetric tessellating pattern may be rare and thus potentially impressive. A more rigorous metric, although more computationally expensive, is recreation effort, i.e. the amount of effort required to design a similar artifact from scratch. The rarity heuristic, which is computationally easier to apply, was shown to largely agree with the recreation effort heuristic, and is therefore used in this paper [11].

While such algorithmic heuristics may not always agree with human intuition about how impressive an artifact is, they may still provide an automatic means to generate an interesting diversity of artifacts. This observation motivates applying the impressiveness metrics in the experiments in this paper in hopes of seeding search to more economically leverage human input. That is, in some experiments users from AMT are presented purely computationally-evolved impressive artifacts in the first generation. In particular, such experiments in this paper explore the same image evolution domain as Lehman and Stanley [11] although impressiveness metrics could be adapted to other domains, i.e. the general concepts are not particular to evolving images.



**Fig. 1. Comparing random and impressive images.** The images shown in (a),(b), and (c) are representative of images generated by random genomes in the image evolution domain, while the images shown in (d),(e), and (f) are examples of images evolved through the impressiveness metrics. Importantly, the impressive images differ qualitatively and are noticeably more complex than the random images.

To facilitate measuring impressiveness in the image evolution domain, Lehman and Stanley [11] compiled a list of features relevant to human vision, such as the level of symmetry in an image, an image’s brightness, and how compressible an image was by different compression algorithms. Next, the rarity of different settings of simple combinations of these features was measured by sampling a space of images [17]. Finally, a search process driven to explore combinations of features was run, and its products screened by the rarity heuristic to cull the most impressive. The setup is described more exhaustively by Lehman and Stanley [11]. Note that other automated content-generation methods could have been substituted; what is most important for the purpose of this paper is that the impressiveness metrics enable automatic evolution of images that are more interesting than the random genomes that would otherwise seed evolution. For comparison, examples of impressive images and those generated from random genomes are shown in figure 1.



### 3 Approach

While previous approaches to IEC are limited by user fatigue or require that a domain be enjoyable to attract users, the approach in this paper, IEC+HCM, avoids such issues by paying users for performing IEC evaluations through a HCM (figure 2). Of course, the trade-off is that with IEC+HCM there is an explicit economic cost for each evaluation.

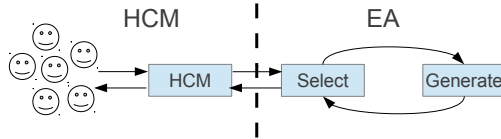
The particular HCM applied here is AMT. Recall that AMT provides a computational interface for posting small computational tasks with a set monetary reward. Importantly, the AMT interface can be applied to automate IEC tasks (e.g. by presenting a user with an artifact or behavior and querying for evaluation via a web interface). Thus, tasks can be mechanically created and uploaded to AMT, results collected, and participants paid, without additional human oversight. In this way the methodology can potentially scale to arbitrary limits given enough money and available users in the HCM, thereby overcoming previous limitations to easily implementing IEC in any domain on a large scale. This paper evaluates the feasibility of the IEC+HCM approach: The experiments included 726 unique AMT users completing 2,300 HIT evaluations. If successful, it should be eventually possible to scale the approach by two to three orders of magnitude.

Importantly, there are many potential ways to combine a HCM with an IEC algorithm. One design decision is how tasks should be divided. For example, a task sent to a HCM for completion could consist of evaluating only a single artifact, evaluating the evolutionary algorithm’s entire population, or guiding multiple generations of IEC evolution (i.e. evaluating multiple populations in sequence). In this paper, tasks were divided into evaluations of a single population each, similarly to how a user influences a single generation of evolution in most single-user IEC applications [19].

Another important decision is what type of input should be gathered from human users; such input could consist of only which artifact in a population was most preferred, or could require individually rating each artifact. Individual ratings were gathered in this paper to enable comparisons between generations and runs, and to encourage greater deliberation during evaluation.

A final aspect of combining IEC with a HCM is how user evaluations of artifacts guide evolution. For example, if each user independently guides evolution for many generations, their most-preferred artifacts could seed the initial population of future tasks, similarly to how sites such as Picbreeder work [17]. However, in the approach in this paper each user evaluates only one generation at a time, and multiple independent evaluations of the same population are combined together to allocate offspring for the next generation. To avoid averaging out individual preferences, children are allocated to artifacts in proportion to how many users rate them most-highly (instead of simply averaging each artifact’s ratings).

Thus while other approaches to IEC+HCM may also be viable, the approach described here reasonably combines IEC with HCMs, and its design decisions form a coherent methodology.



**Fig. 2. The IEC+HCM approach.** During each generation of evolution, when evaluating the population the EA uploads evaluation tasks to the HCM to be completed by human users. The results guide which parents reproduce to form the next generation. In this way, an EA can be driven by the human judgment of many non-experts.

## 4 Experiments

In following sections, experiments are presented that apply IEC+HCM to two domains. Results in evolving aesthetic images are first presented as a domain characteristic of collaborative IEC websites driven by user volunteers [17]. The second domain evolves only compositions of image layouts (which removes the potential for engaging novelty) as an exemplar for where economic incentives are crucial for success.

### 4.1 General Experimental Setup

For all experiments, human computation was purchased through AMT and a set price of \$0.05 USD was paid per user completion of a task. A standard genetic algorithm was applied with a small population size (nine individuals) characteristic of many IEC domains [19]. All runs consisted of ten generations, with one task uploaded per generation, and ten independent runs of each compared method were performed to enable statistical comparisons.

Tasks uploaded to AMT contained nine images, i.e. the entire population, and required users to rate each image’s aesthetic appeal on an integer scale from one to five (where five is the best). Because aesthetic judgment is subjective and varies between individuals, each task was evaluated by five separate AMT users to get a more representative sample. In particular, individuals were selected proportionally to how many users (from the five total) rated them most highly among the nine presented images (i.e. only a user’s highest-rated images would contribute to selection). Preliminary experiments demonstrated that simply averaging the ratings was less effective than this approach because many interesting artifacts were divisive, i.e. they were rated high by some and low by others, and averaging thereby would have resulted in selection for mediocre images that received lukewarm ratings from most users.

### 4.2 Evolving Aesthetic Images with IEC+HCM

The first two experiments explored an image evolution domain that implements an encoding similar to the Picbreeder collaborative IEC website [17] and explored

elsewhere in single-user IEC applications [18]. In particular, in these systems images are represented by ANN-like networks called CPPNs, which are briefly reviewed in the next paragraph (a more detailed introduction is given by Stanley [18]).

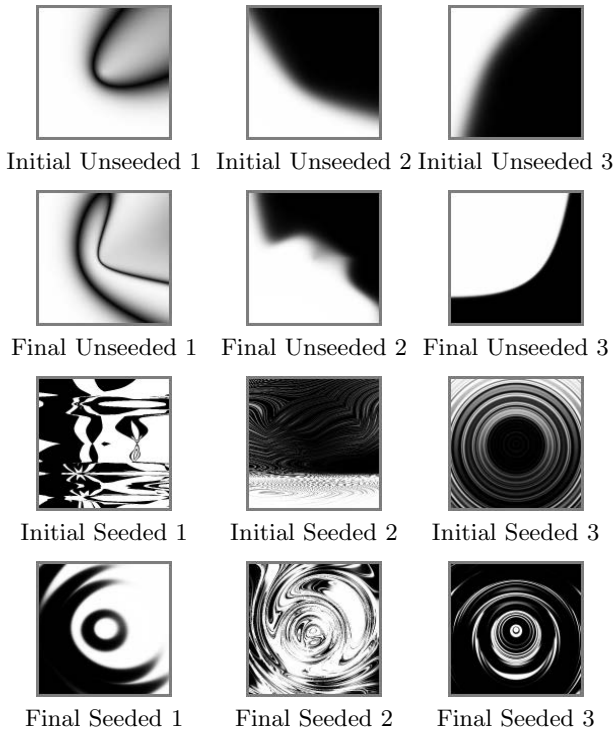
Importantly for the second experiment in this domain, while each node in a traditional ANN is typically the same sigmoid activation function, each node in a CPPN has an activation function selected from a fixed *set* of such functions; the motivation for such a set is to enable more interesting visual patterns through deliberate choice of included functions. For example, sinusoidal functions may be included to enable repetitive patterns and Gaussian functions may be included to enable symmetric ones. In this domain, a CPPN is mapped to the image it represents in the following way: For each pixel in an image, the CPPN’s inputs are set to its scaled Cartesian coordinates, and the output of the network is interpreted as a grayscale pixel value. In effect, the CPPN thus represents a pattern over a coordinate space, which in this case is interpreted as a picture.

The first experiment in this image evolution domain, which is described next, explores methods for seeding newly-created collaborative IEC websites with content.

**Experiment 1: Bootstrapping Collaborative IEC.** The goal of this experiment is to show that IEC+HCM can be applied to evolve aesthetic images through selection purchased from a diffuse cloud of users. One practical application of such a technique is to *bootstrap* newly launched collaborative IEC websites with initial content. For example, for a site like Picbreeder to attract users, it helps to first have a diversity of aesthetic images that users can explore and interact with. Problematically, such initial content is difficult to produce automatically, because aesthetic evaluation generally requires human judgment. On the other hand, it is laborious and uninteresting for humans to generate because initially random images are of poor quality. Therefore, paid users are instrumental for generating such initial content with sufficient quality.

However, because IEC+HCM incurs a financial cost for each evaluation, it becomes important to leverage human input as efficiently as possible. Thus a promising approach to increase IEC+HCM’s financial viability is to first generate a diversity of content *algorithmically*. Such content is more interesting than the random genomes that would otherwise seed evolution, although still in need of further human refinement.

Thus, to investigate this idea two versions of IEC+HCM are run: one method that is first seeded with pre-evolved impressive images, and another that is instead initialized with random genomes. For the unseeded runs, evolution starts from simple random CPPNs in the same way as most other CPPN-encoded image evolution applications [17,18]. For the seeded runs, the setup of Lehman and Stanley [11] was applied to first evolve impressive artifacts, of which the most impressive from 20 separate runs were sampled to seed evolution. Figure 3 shows the representative products of both methods.

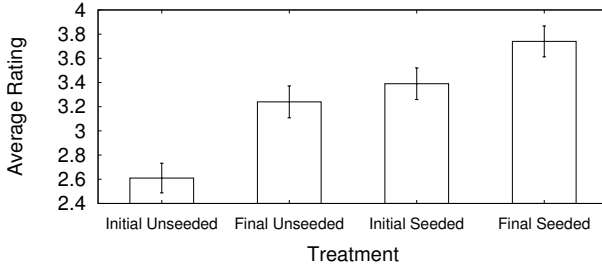


**Fig. 3. Typical Products of the Seeding Experiment.** Images from three representative runs of the seeded and unseeded methods are shown. The labels indicate whether the images are the most-preferred images from the initial or the final generation, and whether they are from the seeded or unseeded method; the number distinguishes separate runs. The main results are (1) that there is a large difference in complexity and quality between the unseeded and seeded runs, and (2) that for the seeded runs there is a noticeable divergence between the initially-preferred seed and the final most-preferred evolved image.

Because judging aesthetic appeal requires subjective human evaluation, AMT was also applied to investigate the products of the two methods. In particular, runs of each method were first arbitrarily paired for comparison. Then, the most preferred images from the initial and final generations of both methods were placed in random order and uploaded to the same AMT evaluation task used for IEC, but with a larger number of separate user evaluations (ten instead of five).

The results of this evaluation process (seen in figure 4) show that the most preferred “impressive” seed images from the first generation of the seeded runs are rated significantly more aesthetically pleasing than are the first generation images from the unseeded runs (Mann-Whitney U-test;  $p < 0.05$ ), supporting their motivation. Furthermore, the champion of the final generation of the seeded runs (i.e. the most-preferred product of human elaboration of the seed images) is rated significantly more pleasing than both the initial generation of

the seeded runs and the final generation of the unseeded runs ( $p < 0.05$ ). In this way, the results support the hypotheses that IEC+HCM can be leveraged to evolve increasingly aesthetically pleasing artifacts and that seeding IEC+HCM with pre-evolved artifacts can more efficiently leverage human evaluations. Thus seeded IEC+HCM may prove a viable technique for bootstrapping collaborative IEC websites.



**Fig. 4. Seeding experiment evaluation.** An independent evaluation comparing the champions of the first and final generations of both the unseeded and seeded methods is shown. The main result is that the final seeded champions are on average rated significantly more aesthetically pleasing than both the final unseeded and initial seeded images.

**Experiment 2: Validating Components of an IEC System.** The next experiment is motivated by the desire to make principled design decisions while creating a collaborative IEC website or single-user IEC application. That is, it is difficult for a system designer to decide objectively on appropriate parameter settings or what genetic encoding is best, especially when the quality of such decisions depends upon subjective factors aggregated across all targeted users (e.g. the aesthetic quality of artifacts evolved under such a decision). The problem is that the most readily available heuristic for the designer, i.e. his own aesthetic preferences or those of his team, may not well reflect the broader preferences of the average target user.

While single-user IEC applications are easy to revise from user feedback even after they have been first released, launching an IEC website inherently involves a certain level of commitment to the domain. That is, changing the domain after the website has launched may invalidate already-evolved content, potentially alienating users whose creative products are deleted. Additionally, if previous content cannot be merged into the system after a domain or encoding changes, then its loss will make the site as a whole less engaging. Therefore it is desirable to avoid such problems and launch a better initial product.

A potential solution is to run controlled experiments with IEC+HCM to collect empirical evidence of a change’s impact from a representative sample of potential users. That is, the quality of results from IEC with different parameter settings or features can be compared, by paying users through AMT to perform selection and then by paying other users to compare the final results. In this

way, the particular preferences of the system’s designers need not dominate the design of the system itself.

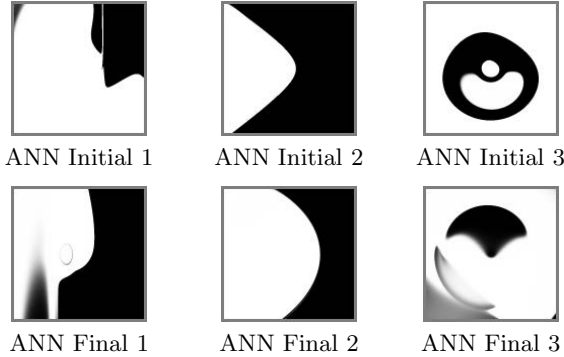
As a simple example, the second experiment evaluates the hypothesis that the additional activation functions of CPPNs improve the aesthetic quality of CPPN-evolved images beyond the use of simpler ANNs [18,17]. A third version of the image evolution task was devised with simple ANNs (i.e. standard ANNs with a single sigmoid activation function) substituted for CPPNs (which have an extended set of activation functions). Only the activation function set is varied, all other aspects of the encoding remain unchanged. In this way, the aesthetic quality of products evolved with CPPNs could be compared to those evolved with simpler ANNs. Furthermore, taking into account the advantages of seeded IEC+HCM runs demonstrated in the previous experiment, only a seeded method with simple ANNs was considered. Note that what is impressive or rare in a particular genetic space depends upon the encoding; therefore, to accomplish seeding with these ANNs required evolving impressive artifacts in this different genetic space. Thus, ten additional IEC+HCM runs were conducted, with ANNs seeded with impressive ANNs in the same way as in the previous experiment with CPPNs.

The effect of replacing CPPNs with ANNs on the results of IEC+HCM is shown in figure 5. As expected, these images differ noticeably from the previous results with CPPNs shown in figure 3. An empirical investigation of the aesthetic difference between the seeded IEC+HCM methods with CPPN and ANNs was then conducted similarly to the previous experiment: AMT users compare the products from paired runs of this simple ANN method (shown in figure 5) and the previous IEC+HCM method with CPPNs (i.e. the final seeded images from figure 3). Figure 6 shows the results: Expanding the set of activation functions in CPPNs facilitates evolving more aesthetically pleasing images. This result demonstrates how IEC+HCM enables objective investigations of the impact of different system features.

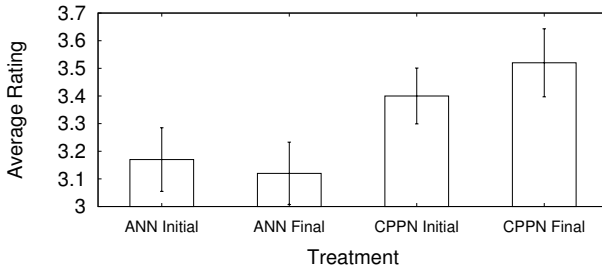
### 4.3 Experiment 3: Evolving Image Layouts with IEC+HCM

The third experiment investigates whether IEC+HCM can expand the range of domains where large-scale IEC can be effectively applied. While large-scale IEC is currently applicable only to domains that are sufficiently enjoyable to attract volunteer users, the IEC+HCM approach can potentially be applied to any domain regardless of how engaging it is, and can be scaled to the extent that funds are available to do so.

Interestingly, the previous experiments with evolving images provide tentative evidence for this hypothesis: The IEC+HCM setup still produced aesthetic improvements in evolved images even though it perverts what is typically enjoyable about such domains, i.e. as a means for expressing personal creativity [19,17]. That is, while in most IEC image evolution systems the user intentionally and



**Fig. 5. Typical Products of the Seeded ANN Runs.** Images are shown from three representative runs of seeded IEC+HCM with ANNs (instead of with CPPNs as in the previous experiment). The qualitative difference between these images and those evolved with CPPNs (figure 3) suggests that the added activation functions of CPPNs impact the kind of images likely to be evolved.



**Fig. 6. Feature Validation Experiment Evaluation.** An independent evaluation comparing the initial generation and final generation champions of seeded runs with the ANN method (ANN initial and ANN final) with those with of the CPPN method (CPPN initial and CPPN final) are shown. The main result is that the final generation CPPN images are judged significantly more aesthetically pleasing than either of the two classes of ANN images (Mann-Whitney U-test;  $p < 0.05$ ). The conclusion is that CPPNs facilitate evolving more aesthetic images than ANNs.

directly drives the creative process through selection, with IEC+HCM a particular human’s input is only a transient interchangeable part of a larger process, diluting any overarching influence on evolution. Supporting this idea, of the 620 unique AMT users who contributed to the image evolution IEC+HCM experiments, 419 users completed only *one* evaluation task (i.e. they interacted with the system only once and could not have seen any effect of their influence), and only 65 users contributed more than four evaluations. Thus it is unlikely that any particular user will receive the satisfaction of seeing their aesthetic influence realized.

However, image evolution still offers the potential of novelty between evaluations, which may be interesting for a user even if IEC+HCM does not allow for directly expressing creativity as in other IEC systems. Thus to more directly test the hypothesis that IEC+HCM can extend the reach of large-scale IEC to domains not inherently enjoyable, the third experiment explores an intuitively less enjoyable task, that of evolving the layout of an image composition. In particular, the task is to evolve the relative positions of a fixed set of images (seen in figure 7) to maximize the aesthetic appeal of the composition. Unlike the image evolution domain, the potential for novelty is limited because the components of the image are always the same and uninteresting.

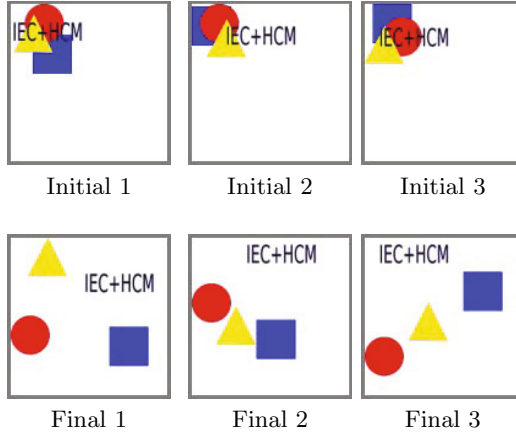
The domain and genetic encoding are illustrated by figure 7. Note that the same IEC+HCM setup as previously described was adapted for this third experiment but with only a single unseeded method. While seeding with impressive pre-evolved layouts might accelerate progress in this domain, such seeding is not necessary to verify the hypothesis.



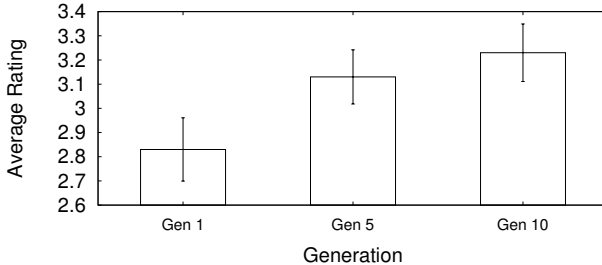
**Fig. 7. The Image Layout Domain.** The image layout experiment evolves a composition of the four shown images through IEC+HCM. The encoding is a simple list of Cartesian coordinates that specify the offset of each of each image. Initial genomes are generated such that they cluster the four images near the upper left corner, providing a predictably poor starting arrangement for human-guided evolution to improve upon. Mutation perturbs the coordinates of one image out of the four, adding to the  $x$  and  $y$  coordinate a separately number chosen uniformly between  $-50$  and  $50$ .

The products of this experiment are shown in figure 8. The results were validated similarly to the previous experiments, by presenting them to be rated by a larger set of AMT users. However, instead of comparing between methods, evolved artifacts are compared over generations. The idea is to demonstrate that progress in aesthetic evolution is occurring. The aggregated ratings from the larger validation evaluation are shown in figure 9. As expected, the most-preferred layouts from the final generation are rated significantly more pleasing in appearance than those from the first generation, thus supporting the conclusion that evolutionary progress was facilitated by IEC+HCM in this domain. Note that while the domain itself is somewhat trivial, the results provide an existence proof that IEC+HCM can extend large-scale IEC to domains that are not inherently enjoyable.





**Fig. 8. Typical Products of the Layout Evolution Experiments.** Images are shown from three representative runs of IEC+HCM in the layout evolution domain. In particular, the most-preferred image from the initial and final generation of the runs are shown. Over evolution, the images composing the layouts expand to better fill the space. The conclusion is that IEC+HCM can be successfully applied even in domains that are not inherently enjoyable.



**Fig. 9. Image Layout Experiment Evaluation.** An independent evaluation comparing the champions of the first, fifth, and final (tenth) generations from the ten independent IEC+HCM runs in the image layout domain is shown. The main result is that the image layout of final champions is judged significantly more aesthetic than that of the first generation (Mann-Whitney U-test;  $p < 0.05$ ).

## 5 Discussion and Future Work

This paper investigated leveraging markets for human computation to support large-scale IEC in three ways. Exploratory experiments in this paper showcase how the ability to pay for human computation potentially can bootstrap IEC websites, inform the design of such websites or single-user IEC systems, and act as a viable alternative to such websites when the domain is not enjoyable or engaging.

In this way, an interesting advantage of the IEC+HCM approach is that it bypasses the significant problem of user fatigue in IEC [19] through paying users directly, without constraining the domain. Of course, the trade-off is that pairing IEC with human computation incurs an explicit financial cost per evaluation. While such financial costs have always been implicit in strictly computational EC (e.g. costs to maintain a cluster of computers necessary for large-scale experimentation) and collaborative IEC websites (e.g. server costs), the price of human computation does not similarly benefit from Moore's law. Thus large-scale IEC+HCM may be most applicable for unengaging domains limited by difficulty in applying appropriate selection pressure, and also possibly for commercial applications where the cost of IEC+HCM is less than the value of the evolved artifact.

So while the IEC+HCM mechanism can be leveraged to improve the design and engagement of single-user IEC systems and collaborative IEC websites, its most interesting implication may be that exploiting it on a large scale may potentially lead to results exceeding current approaches in evolutionary robotics or artificial life. That is, to the extent that current approaches are limited by lack of appropriate selection pressure [25,14,10], and to the extent that human judgment can remedy such limitations [5,22], human computation may be a technique that can be exploited to further the state of the art in EC. For example, large-scale IEC+HCM with a significant budget applied to evolving virtual creatures might produce creatures with complexity and functionality beyond the reach of current methods. In this way, an interesting direction for further experimentation is to apply IEC+HCM to evolve controllers for evolutionary robotics or artificial life experiments.

## 6 Conclusion

This paper explored combining interactive evolution with human computation markets to purchase a powerful form of selection pressure. The promise of the approach was shown in preliminary experiments evolving aesthetic images and the layout of image compositions. Applying the same techniques in other domains limited by lack of appropriate selection pressure may enable evolution of more complex artifacts or behaviors than previously possible. The conclusion is that human computation markets may be an important tool for supporting collaborative IEC websites as well as for extending the reach of large-scale IEC beyond only task domains that are enjoyable.

## References

1. Chou, C., Kimbrough, S., Sullivan-Fedock, J., Woodard, C., Murphy, F.: Using interactive evolutionary computation (iec) with validated surrogate fitness functions for redistricting. In: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference, pp. 1071–1078. ACM (2012)

2. Clune, J., Lipson, H.: Evolving three-dimensional objects with a generative encoding inspired by developmental biology. In: Proceedings of the European Conference on Artificial Life (2011)
3. den Heijer, E., Eiben, A.E.: Comparing aesthetic measures for evolutionary art. In: Di Chio, C., Brabazon, A., Di Caro, G.A., Ebner, M., Farooq, M., Fink, A., Grahl, J., Greenfield, G., Machado, P., O'Neill, M., Tarantino, E., Urquhart, N. (eds.) *EvoApplications 2010, Part II*. LNCS, vol. 6025, pp. 311–320. Springer, Heidelberg (2010)
4. Goldberg, D.E.: Simple genetic algorithms and the minimal deceptive problem. In: Davis, L.D. (ed.) *Genetic Algorithms and Simulated Annealing*, Research Notes in Artificial Intelligence. Morgan Kaufmann (1987)
5. Gruau, F., Quatramaran, K.: Cellular encoding for interactive evolutionary robotics. In: Fourth European Conference on Artificial Life, pp. 368–377. MIT Press (1997)
6. Hastings, E., Stanley, K.: Interactive genetic engineering of evolved video game content. In: Proceedings of the 2010 Workshop on Procedural Content Generation in Games, p. 8. ACM (2010)
7. Ipeirotis, P.: Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads. The ACM Magazine for Students* 17(2), 16–21 (2010)
8. Kittur, A., Chi, E.H., Suh, B.: Crowdsourcing user studies with mechanical turk. In: Proceedings of the Twenty-sixth Annual SIGCHI Conference on Human Factors in Computing Systems, pp. 453–456. ACM (2008)
9. Kosorukoff, A.: Human based genetic algorithm. In: 2001 IEEE International Conference on Systems, Man, and Cybernetics, pp. 3464–3469. IEEE (2001)
10. Lehman, J., Stanley, K.O.: Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation* 19(2), 189–223 (2011)
11. Lehman, J., Stanley, K.O.: Beyond open-endedness: Quantifying impressiveness. In: Proceedings of Artificial Life Thirteen, ALIFE XIII (2012)
12. MacCallum, R., Mauch, M., Burt, A., Leroi, A.: Evolution of music by public choice. *Proceedings of the National Academy of Sciences* 109(30), 12081–12086 (2012)
13. McCormack, J.: Open problems in evolutionary music and art. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) *EvoWorkshops 2005*. LNCS, vol. 3449, pp. 428–436. Springer, Heidelberg (2005)
14. Miconi, T., Channon, A.: An improved system for artificial creatures evolution. In: Proceedings of the Tenth International Conference on Artificial Life (ALIFE X), pp. 255–261. MIT Press (2006)
15. Orkin, J., Roy, D.: The restaurant game: Learning social behavior and language from thousands of players online. *Journal of Game Development* 3(1), 39–60 (2007)
16. Risi, S., Lehman, J., D'Ambrosio, D.B., Hall, R., Stanley, K.O.: Combining search-based procedural content generation and social gaming in the petalz video game. In: Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2012) (2012)
17. Secretan, J., Beato, N., D'Ambrosio, D., Rodriguez, A., Campbell, A., Folsom-Kovarik, J., Stanley, K.: Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evol. Comp.* (2011) (to appear)
18. Stanley, K.: Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines* 8(2), 131–162 (2007)

19. Takagi, H.: Interactive evolutionary computation: Fusion of the capacities of EC optimization and human evaluation. *Proceedings of the IEEE* 89(9), 1275–1296 (2001)
20. Von Ahn, L.: Games with a purpose. *Computer* 39(6), 92–94 (2006)
21. Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., Blum, M.: Recaptcha: Human-based character recognition via web security measures. *Science* 321(5895), 1465–1468 (2008)
22. Woolley, B.G., Stanley, K.O.: Exploring promising stepping stones by combining novelty search with interactive evolution. *CoRR* abs/1207.6682 (2012)
23. Yu, L., Nickerson, J.: An internet-scale idea generation system. *ACM Transactions on Interactive Intelligent Systems* 3(1), 2013 (2011)
24. Yu, L., Nickerson, J.V.: Cooks or cobblers?: crowd creativity through combination. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1393–1402. ACM (2011)
25. Zaera, N., Cliff, D., Bruten, J.: (Not) evolving collective behaviours in synthetic fish. In: *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*. MIT Press Bradford Books (1996)

# A New Version of the Multiobjective Artificial Bee Colony Algorithm for Optimizing the Location Areas Planning in a Realistic Network

Víctor Berrocal-Plaza\*, Miguel A. Vega-Rodríguez, and Juan M. Sánchez-Pérez

Dept. of Computers & Communications Technologies, University of Extremadura  
Escuela Politécnica, Campus Universitario S/N, 10003, Cáceres, Spain  
{vicberpla,mavega,sanperez}@unex.es

**Abstract.** In this paper, we present our version of the MultiObjective Artificial Bee Colony algorithm (a metaheuristic based on the foraging behavior of honey bees) to optimize the Location Areas Planning Problem. This bi-objective problem models one of the most important tasks in any Public Land Mobile Network: the mobile location management. In previous works of other authors, this management problem was simplified by using the linear aggregation of the objective functions. However, this technique has several drawbacks. That is the reason why we propose the use of multiobjective optimization. Furthermore, with the aim of studying a realistic mobile environment, we apply our algorithm to the mobile network developed by the Stanford University (a mobile network located in the San Francisco Bay, USA). Experimental results show that our proposal outperforms other algorithms published in the literature.

**Keywords:** Location Areas Planning Problem, Mobile Location Management, Multiobjective Optimization, Stanford University Mobile Activity Traces, Artificial Bee Colony Algorithm.

## 1 Introduction

The Public Land Mobile Networks (PLMNs) are the networks that provide mobile communications to the public. For it, the desired coverage area is divided into several smaller regions (known as network cells) with the aim of providing service to a huge number of mobile subscribers with few radioelectric resources [1]. In such networks, the mobile location management is a fundamental task whereby the mobile network tracks the subscribers' movement with the goal of redirecting the incoming calls to the callee users.

Commonly, the mobile location management consists of two main procedures: the subscriber location update (*LU*), and the paging (*PA*) [9]. The *LU* is initiated by the mobile station (MS, the subscriber terminal) according to a predefined method. There are several *LU* procedures published in the literature: never

---

\* The work of Víctor Berrocal-Plaza has been developed under the Grant FPU-AP2010-5841 from the Spanish Government. This work was partially funded by the Spanish Ministry of Economy and Competitiveness and the ERDF (European Regional Development Fund), under the contract TIN2012-30685 (BIO project).

update, whenever the MS moves to a new cell (or always update), whenever the MS moves to a new area (location areas), every certain number of visited cells (distance-based), every certain number of cell crossing (movement-based), etc. [16]. However, the location update based on location areas highlights as one of the most popular strategies in current mobile networks, and therefore, this is the *LU* procedure studied in this work. On the other hand, the *PA* procedure is initiated by the mobile network to know the exact cell in which the callee subscribers are located [13]. For it, several paging messages are broadcast around the last known location. This last procedure can be classified into two main groups: simultaneous paging and sequential paging. In the simultaneous paging, the network cells are simultaneously polled. And in the sequential paging, the network cells are grouped into paging areas that are sequentially polled. Due to the time constraint associated with this last procedure, the maximum number of paging cycles (and hence, the maximum number of paging areas) is restricted to be lower or equal to three [13].

In this work, we research the mobile location management based on location areas with the two-cycle sequential paging defined in [15]. This *LU* strategy groups the network cells into continuous and non-overlapped logical areas (or Location Areas, LAs [16]) with the aim of delimiting the signaling load associated with the *LU* and *PA* procedures. In this way, the mobile station only updates its location whenever it moves to a new Location Area (LA), and the paging procedure is only performed within the last updated LA. Therefore, the proper dimensioning of this location management strategy is an important engineering issue, in which the main goal is to find the configurations of LAs that simultaneously reduce the number of location updates and the number of paging messages. That is, the Location Areas Planning Problem (LAPP) is a multiobjective optimization problem with two objective functions (as we will see in Section 3).

With the aim of finding the best possible configurations of location areas, we propose a new version of the MultiObjective Artificial Bee Colony algorithm (MO-ABC, a swarm intelligence algorithm based on the foraging behavior of honey bees [14]). Furthermore, in order to study a realistic mobile environment, we solve the mobile network developed by the Stanford University [11], because it is well-validated against real data measured in the San Francisco Bay (USA).

The rest of the paper is organized as follows. Section 2 presents the related works. The Location Areas Planning Problem is defined in Section 3. The main features of a multiobjective optimization problem, the quality indicators used in this work, and a detailed explanation of our proposal are shown in Section 4. Experimental results and comparisons with other works are presented in Section 5. Finally, Section 6 discusses our conclusions and future work.

## 2 Related Work

The mobile location management based on location areas has been widely researched in the last decade due to the exponential increase in the number of

mobile subscribers. There are several works in the literature in which different metaheuristics have been applied to optimize the Location Areas Planning Problem. P. R. L. Gondim in [10] was one of the first authors to define the LAPP as an NP-hard combinatorial optimization problem due to the huge size of the objective space. In his work, he proposed a Genetic Algorithm (GA) to find quasi-optimal configurations of Location Areas. P. Demestichas et al. in [8] proposed three metaheuristics (Simulated Annealing (SA), Tabu Search (TS), and GA) to study the LAPP in different mobile environments. R. Subrata and A. Y. Zomaya proposed a dynamic location update procedure based on location areas [15], which allows reducing the number of location updates, but not the total signaling load because a personalized configuration of location areas should be calculated, stored, and transmitted whenever a mobile station moves out of its current location area. In this work, R. Subrata and A. Y. Zomaya studied the real-time mobile activity trace developed by the Stanford University [11], a mobile activity trace that is well-validated against real data measured in the San Francisco Bay (USA). This mobile activity trace was also studied in [2,3], where S. M. Almeida-Luz et al. optimized the LAPP with the algorithms: Differential Evolution (DE, [2]) and Scatter Search (SS, [3]).

In all of these related works, the LAPP was solved by using different algorithms from the single-objective optimization field. For it, the two objective functions of the LAPP were linearly combined into a single objective function. This technique allows simplifying the optimization problem, but has several drawbacks (see Section 3). That is why our research focuses on optimizing this problem with different multiobjective metaheuristics. Recently, we have just published our versions of two well-known MultiObjective Evolutionary Algorithms (MOEAs): the Non-dominated Sorting Genetic Algorithm II (NSGAI2 [5]), and the Strength Pareto Evolutionary Algorithm 2 (SPEA2 [4]).

### 3 Location Areas Planning Problem

The location update procedure based on location areas (by definition, a location area is a continuous and non-overlapped group of network cells) is being widely used in current Public Land Mobile Networks. In this location update strategy, a mobile station is free to move inside a given location area without updating its location, and the paging procedure is only performed in the network cells within the last updated location area (because the network knows the subscriber location at a location area level). Therefore, the main challenge of the location areas scheme is to find the configurations of location areas that minimize signaling load associated with the location update procedure ( $LU_{cost}$ ) and the paging load ( $PA_{cost}$ ). In this work, we use a two-cycle sequential paging based on the last known location (last updated cell), the same paging procedure is used in [15,2,3,5]. Formally, these two objective functions can be expressed as Equation 1 and Equation 2 respectively, where:  $\gamma_{t,i}$  is a binary variable that is equal to 1 when the mobile station  $i$  moves out of its current location area in the time  $t$ , otherwise  $\gamma_{t,i}$  is equal to 0.  $[T_{ini}, T_{fin}]$  is the time interval of the mobile activity

trace.  $N_{user}$  is the number of mobile users.  $\rho_{t,i}$  is a binary variable that is equal to 1 only when the subscriber  $i$  has an incoming call in the time  $t$ .  $\alpha_{t,i}$  is a binary variable that is equal to 1 only when the mobile station  $i$  is located in its last updated cell in the time  $t$ .  $NA$  is a vector that stores the number of network cells of each location area. And  $LA_t$  is a vector that stores the location area in which every mobile subscriber is located in the time  $t$ .

$$f_1 = \min \left\{ LU_{cost} = \sum_{t=T_{ini}}^{T_{fin}} \sum_{i=1}^{N_{user}} \gamma_{t,i} \right\}, \quad (1)$$

$$f_2 = \min \left\{ PA_{cost} = \sum_{t=T_{ini}}^{T_{fin}} \sum_{i=1}^{N_{user}} \rho_{t,i} (\alpha_{t,i} + \bar{\alpha}_{t,i} \cdot NA[LA_t[i]]) \right\}. \quad (2)$$

It should be noted that these two objective functions are conflicting. That is, if we would reduce the signaling load of the location update procedure, we should increase the size of the location areas, which leads to an increment in the number of paging messages because a larger number of network cells have to be polled whenever a subscriber has an incoming call. And, on the other hand, if we reduce the paging cost with smaller location areas, we will have more location updates (increasing this other cost).

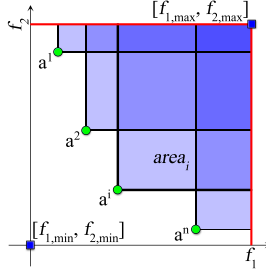
In previous works from other authors, this optimization problem was simplified by using the linear aggregation of the objective functions (see Equation 3). However, this technique has several drawbacks. Firstly, a very accurate knowledge of the problem is required to properly configure the weight coefficients ( $\alpha, \beta \in \mathfrak{R}$ ). Secondly, different states of the signaling network might require of different values of  $\alpha$  and  $\beta$ . And thirdly, a single-objective optimizer must perform an independent run for each combination of the weight coefficients. That is why we propose the use of multiobjective metaheuristics to optimize the LAPP. Furthermore, a multiobjective optimization algorithm provides a wide range of solutions among which the network operator could select the one that best adjusts to the real state of the signaling network.

$$f_{SOA}(\alpha, \beta) = \alpha \cdot f_1 + \beta \cdot f_2. \quad (3)$$

## 4 Multiobjective Optimization Paradigm

A Multiobjective Optimization Problem (MOP) can be defined as an optimization problem in which two or more conflicting objective functions have to be optimized simultaneously under certain constraints [6] (e.g. the Location Areas Planning Problem). In the following and without loss of generality, we assume a minimization MOP with two objective functions (as the LAPP). In this kind of problems, the main challenge consists in finding the best possible set of solutions, where each solution is related to a specific trade-off among objectives. These solutions are referred as non-dominated solutions, and the set of





**Fig. 1.** Hypervolume for a minimization problem with two objectives

non-dominated solutions is commonly known as Pareto Front. For definition, a solution  $\mathbf{x}^1$  is said to dominate the solution  $\mathbf{x}^2$  (denoted as  $\mathbf{x}^1 \prec \mathbf{x}^2$ ) when  $\forall i \in [1, 2], f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2) \wedge \exists i \in [1, 2] : f_i(\mathbf{x}^1) < f_i(\mathbf{x}^2)$ .

In the literature, there are several indicators to measure the quality of a Pareto Front in the multiobjective context. The quality indicators used in this work are presented in Section 4.1 to Section 4.3. Section 4.4 presents a detailed explanation of our version of the MultiObjective Artificial Bee Colony algorithm.

#### 4.1 Hypervolume: $I_H(A)$

In a bi-objective MOP, the Hypervolume indicator ( $I_H(A)$ ) computes the area of the objective space that is dominated by the Pareto Front  $A$ , and is bounded by the reference points (points that are calculated by using the maximum and minimum values of each objective function) [6]. Due to the fact that the main goal of a multiobjective optimization algorithm is to find a wide range of non-dominated solutions, the Pareto Front  $A$  is said to be better than the Pareto Front  $B$  when  $I_H(A) > I_H(B)$ . Fig. 1 shows the  $I_H(A)$  calculation, which can be formally described by Equation 4.

$$I_H(A) = \left\{ \bigcup_i area_i \mid \mathbf{a}^i \in A \right\}. \quad (4)$$

#### 4.2 Set Coverage: $SC(A, B)$

The Set Coverage indicator ( $SC(A, B)$ ) computes the proportion of solutions of the Pareto Front  $B$  that are weakly dominated by the solutions of the Pareto Front  $A$  [6]. For definition, a solution  $\mathbf{x}^2$  is weakly dominated by another solution  $\mathbf{x}^1$  (denoted as  $\mathbf{x}^1 \preceq \mathbf{x}^2$ ) when  $\forall i \in [1, 2], f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2)$ . This indicator establishes that the Pareto Front  $A$  is better than the Pareto Front  $B$  when  $SC(A, B) > SC(B, A)$ . The  $SC(A, B)$  can be expressed as Equation 5, where

the operator  $|\cdot|$  represents the number of solutions of a Pareto Front, or the number of solutions that satisfy a given condition.

$$SC(A, B) = \frac{|\{\mathbf{b} \in B; \exists \mathbf{a} \in A : \mathbf{a} \preceq \mathbf{b}\}|}{|B|}. \quad (5)$$

### 4.3 $\epsilon$ -Indicator: $I_\epsilon(A, B)$

The  $\epsilon$ -Indicator ( $I_\epsilon(A, B)$ ) calculates the minimum distance ( $\epsilon$ ) that the Pareto Front  $B$  must be translated in the objective space to be weakly dominated by the Pareto Front  $A$  [18]. With this indicator, the Pareto Front  $A$  will be better than the Pareto Front  $B$  when  $I_\epsilon(A, B) < I_\epsilon(B, A)$ . The  $I_\epsilon(A, B)$  can be formally represented as Equation 6.

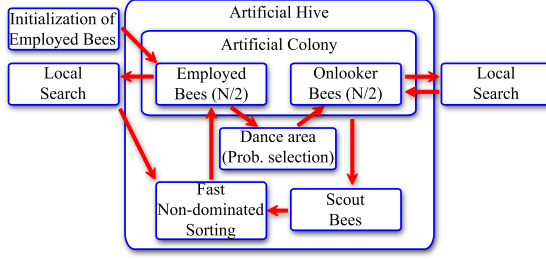
$$I_\epsilon(A, B) = \min \{\epsilon \in \mathbb{R} | \forall \mathbf{b} \in B \exists \mathbf{a} \in A : \forall i \in [1, 2], f_i(\mathbf{a}) \leq \epsilon \cdot f_i(\mathbf{b})\}. \quad (6)$$

### 4.4 Our Multiobjective Artificial Bee Colony Algorithm

In this work, we propose our version of the MultiObjective Artificial Bee Colony algorithm to optimize the Location Areas Planning Problem in a realistic mobile environment. This algorithm was proposed by A. Rubio-Largo et al. in [14] as a multiobjective adaptation of the original Artificial Bee Colony algorithm (ABC), which was proposed by D. Karaboga and B. Basturk in [12]. Basically, the MO-ABC is the original ABC with a *MOFitness* function (used to know the quality of a solution in the multiobjective context), and with the *fast non-dominated sorting* procedure of the well-known Non-dominated Sorting Genetic Algorithm II [7] (used to arrange the population at the end of each generation).

The ABC is a swarm intelligence algorithm based on the foraging behavior of honey bees. In this algorithm, we can distinguish three different entities: the food sources, the artificial colony, and the artificial hive. Every food source represents a possible solution of the problem, being its amount of nectar proportional to the solution quality. The artificial colony (with size  $N$ ) is constituted by three kinds of bees: employed, onlooker, and scout bees. The employed bees (the first half of the artificial colony) search and exploit food sources around the artificial hive, and then, they share their knowledge (quality of the food sources) with the onlooker bees (the second half of the artificial colony), which wait in the dance area of the artificial hive. Subsequently, every onlooker bee selects an employed bee (with a probability proportional to the quality of its food source), and it becomes employed bee of such food source. And finally, a bee becomes a scout bee when its food source is overexploited. This kind of bees performs random search around the artificial hive in order to find new unexploited food sources.

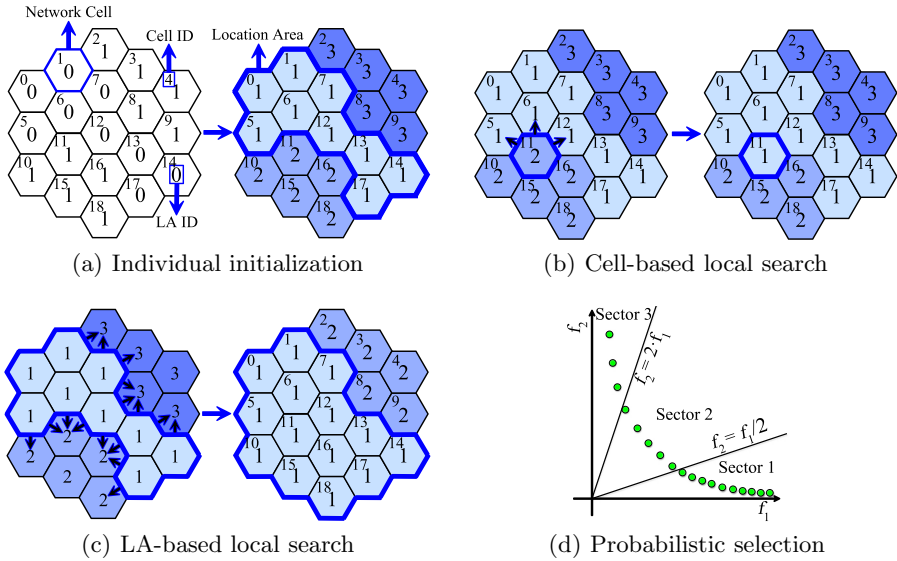
Fig. 2 shows the task decomposition of our version of the MO-ABC algorithm. As we can see in this figure, the first step consists in initializing the first population of employed bees. Note that each bee exploits a food source (a possible solution of the problem), and therefore, a bee can be expressed as an encoded



**Fig. 2.** Task decomposition of our version of the MO-ABC algorithm

solution of the problem. After the initialization process, an iterative method is used to improve this set of initial solutions. In the first step of this iterative method, for each employed bee ( $\mathbf{x}^{EB}$ ), we perform a local search with the aim of finding better solutions ( $\mathbf{x}^{newEB}$ ). Every initial solution is replaced by a new solution when  $\mathbf{x}^{newEB} \prec \mathbf{x}^{EB}$ . In the next step, the *fast non-dominated sorting* [7] is applied to arrange the employed bees according to their quality in the multiobjective context. Subsequently, every onlooker bee selects an employed bee (probabilistic selection of employed bees,  $\mathbf{x}^{OB} \leftarrow \mathbf{x}^{EB}$ ) and performs a local search in order to improve the selected solution. In this case, the selected solution is replaced by a new solution when  $(\mathbf{x}^{newOB} \prec \mathbf{x}^{OB}) \vee (\mathbf{x}^{newOB} \not\prec \mathbf{x}^{OB} \wedge \mathbf{x}^{OB} \not\prec \mathbf{x}^{newOB})$ . With this last condition, we favor the search of new solutions, and even in the same front. The replacement criterion is one of the main differences with the MO-ABC presented in [14], where the *MOFitness* is used to determine if an old solution should be replaced by a new solution. With a criterion based on the dominance concept, we avoid the evaluation of the whole population whenever the local search process finds a new solution. The other differences are in the operators used in our MO-ABC version. The ABC algorithm defines a method to avoid its stagnation. This is done by using the scout bees. For it, every bee has a counter that is incremented whenever its solution is not improved (or replaced) after the local search process. In this way, a bee becomes a scout bee when its counter is higher or equal to a certain number of generations (*limit*). Every scout bee initializes its counter to 0. And finally, the last step of this iterative method consists in selecting the best bees (employed, onlookers, and scouts) as the employed bees of the next generation. For it, we use the *fast non-dominated sorting* to know the quality of every bee (or solution) in the multiobjective context. This iterative method is executed until the stop condition is reached. In this work we use the maximum number of generations as stop condition [2,3,5].

**Individual Initialization.** As mentioned above, a bee is an encoded solution of the problem. In this work, we use a vector representation in which every position of the vector is an integer that represents a network cell and stores the Location Area assigned to that cell. In order to generate the first employed bees, every vector is filled with a random pattern of 0s and 1s (integer numbers). Afterwards,



**Fig. 3.** Operators of our MO-ABC version

this random pattern is used to determine the corresponding configuration of LAs. For it, and according to the definition of Location Area (a continuous and non-overlapped group of cells), every Location Area will be composed by a continuous group of network cells with the same value of vector. Fig. 3(a) shows an example of this procedure.

**Local Searches.** The local searches are used to explore the neighborhood of a solution (in the objective space) with the goal of finding new solutions of greater or equal quality. In this work, we propose two local searches: a cell-based local search, and a LA-based local search. The cell-based local search consists in merging a boundary cell (which is randomly selected) with one of its neighboring LAs. Fig. 3(b) shows an example of this local search. And in the LA-based local search, we merge a randomly selected LA with one of its neighboring LAs (see Fig. 3(c)). In these two procedures, we evaluate all the possibilities and then, we select the one that satisfies the replacement criterion (defined above).

**Probabilistic Selection.** The probabilistic selection is the method used by the onlooker bees to select a solution according to its quality in the multiobjective context. The LAPP has a very important feature, the density of feasible solutions in the objective space decreases when increasing the size of the Location Areas. And therefore, the solutions with high  $LU_{cost}$  (solutions with a high number of small location areas) will be selected with higher probability. With the aim of dealing with such inconvenience, we propose to decompose the normalized

objective space in three sectors (see Fig. 3(d)). Thus, we can directly control the number of solutions selected in each sector. In this work, we define three kinds of onlooker bees, each one specializing in exploiting a specific sector. Furthermore, in order to cause the least possible impact in the search of the algorithm, the number of onlooker bees assigned to each sector is equal to the number of solutions (employed bees) in that sector. On the other hand, a solution within a sector  $S_i$  is selected with probability  $p(\mathbf{x}^{EB,j}) = f_{fit}(\mathbf{x}^{EB,j}) / \sum_{k \in S_i} f_{fit}(\mathbf{x}^{EB,k})$ . The function  $f_{fit}(\mathbf{x})$  is a function that returns a real number, which is equal to the weighted sum of the *Ranking* procedure and the *Crowding distance* [7] ( $f_{fit}(\mathbf{x}) = 10 \cdot Rank(\mathbf{x}) + Crowd(\mathbf{x})$ , only valid for a normalized bi-objective problem). Note that  $f_{fit}(\mathbf{x})$  is proportional to the quality of  $\mathbf{x}$  in the multiobjective context.

**Method for Generating the Scout Bees.** The scout bees are the mechanism that the MO-ABC algorithm uses to avoid its stagnation. In this work, we propose the following way to generate a scout bee. Firstly, we select a sector ( $S_i$ ) with a probability  $p(S_i) = |S_i|^{-1} / \sum_j |S_j|^{-1}$ , where  $|S_i|$  represents the number of solutions in the sector  $S_i$ . Note that this probabilistic selection favors the exploitation of those zones of the objective space with lower number of solutions. Afterwards, we generate a scout bee ( $\mathbf{x}^{SB}$ ) by using Equation 7 (where  $N_{cell}$  is the number of cells in the network), i.e. every scout bee is the average solution of the selected sector.

$$\mathbf{x}_j^{SB} = \frac{1}{|S_i|} \sum_{k \in S_i} \mathbf{x}_j^k, \forall j \in [0, N_{cell} - 1]. \quad (7)$$

## 5 Experimental Results

With the aim of checking the quality of our algorithm in a realistic mobile environment, we study the real-time mobile activity trace developed by the Stanford University [11]. This mobile network is defined in Section 5.1. Furthermore, we compare our proposal with other algorithms published in the literature. This is shown in Section 5.2.

### 5.1 Stanford University Mobile Activity Traces

The Stanford University Mobile Activity TRAcEs (SUMATRA) are a set of test networks that are available via the Internet [11]. In this work, we study the BALI-2 network, a mobile network with 90 cells and 66,550 subscribers. The main appeal of this mobile network is that it corresponds with a real-time mobile activity trace that is well-validated against real data measured in the San Francisco Bay (USA). And therefore, we will be able to analyze the behavior of our algorithm in a realistic mobile environment.

**Table 1.** Comparison with other works

Algorithm	#points	median ( $I_H$ )	iqr ( $I_H$ )	best $f_{\text{SOA}}(10, 1)$
MO-ABC	958	93.90	3.53e-04	2,616,330
NSGAI[5]	772	93.75	1.79e-03	2,619,839
SPEA2[4]	752	92.79	2.02e-03	2,619,697
DE[2]	1	-	-	2,799,289
SS[3]	1	-	-	2,756,836
DBLA[15]	1	-	-	2,695,282

**Table 2.** Comparison among MOEAs:  $SC(PF_A, PF_B)$ 

$PF_A \backslash PF_B$	MO-ABC	NSGAI[5]	SPEA2[4]
MO-ABC	-	66.06	63.43
NSGAI[5]	30.17	-	48.94
SPEA2[4]	26.10	38.21	-

**Table 3.** Comparison among MOEAs:  $I_\epsilon(PF_A, PF_B)$ 

$PF_A \backslash PF_B$	MO-ABC	NSGAI[5]	SPEA2[4]
MO-ABC	-	1.03	1.01
NSGAI[5]	1.24	-	1.01
SPEA2[4]	1.32	1.33	-

## 5.2 Comparison with Other Works

In this section we present a comparison with other algorithms published in the literature [15,2,3,5,4], two of them are our previous multiobjective algorithms (NSGAI [5], and SPEA2 [4]). In order to perform a fair comparison, we have implemented the same local searches in all of our multiobjective algorithms. Furthermore, we use the same population size (300 individuals or bees) and the same stop condition (1000 generations) as the algorithms proposed in [2,3,5,4]. The other parameters of our MO-ABC algorithm have been configured by means of a parametric study of 31 independent runs per experiment. We have selected the parameter configuration that maximizes the  $I_H$  value:  $limit = 15$ ,  $P_{C-LS} = 0.9$ , and  $P_{LA-LS} = 0.1$ , where  $P_{C-LS}$  is the probability of performing the cell-based local search and  $P_{LA-LS}$  is the probability of performing the LA-based local search. In this work, we have defined these two local searches in a way that a bee must only perform one local search per generation.

Table 1 gathers a summary of our comparative study (of 31 independent runs per experiment). In this table we compare our multiobjective algorithms by using statistical data of the  $I_H$  (median and interquartile range), and the number of points of the Pareto Front associated with the median  $I_H$ . In this paper, we use the same reference points as in [5,4]. This comparative study clarifies that our MO-ABC algorithm performs better than our previous multiobjective algorithms because it obtains a higher number of solutions and it achieves a higher  $I_H$  value with a lower interquartile range. The same conclusion can be reached if we compare our algorithms with the Set Coverage (see Table 2) and the  $\epsilon$ -Indicator (see Table 3).

Furthermore, we present a comparison with other approaches published in the literature, all of them Single-objective Optimization Algorithms (SOAs) in which the fitness function is Equation 3 with  $\alpha = 10$  and  $\beta = 1$  [15,2,3]. Regrettably, the best solution found (of a set of runs) is the only data available in these

papers, so we cannot perform a statistical comparison. In order to accomplish this comparison, we have searched in our median Pareto Front the solution that best meets the objective function used by these SOAs (Equation 3 with  $\alpha = 10$  and  $\beta = 1$ ). Table 1 reveals that our MO-ABC is very competitive because it outperforms the SOAs proposed in [2,3] and the Distance-Based Location Area (DBLA) algorithm presented in [15], which is far from trivial because these algorithms [15,2,3] are specializing in finding a single solution.

## 6 Conclusion and Future Work

In this work, we present our version of the MultiObjective Artificial Bee Colony algorithm (MO-ABC, an optimization algorithm based on the foraging behavior of honey bees) to optimize one of the most important planning problems in any Public Land Mobile Network (PLMN): the Location Areas Planning Problem. Furthermore, with the aim of checking the quality of our algorithm in a realistic mobile environment, we study the real-time mobile activity trace developed by the Stanford University [11]. The main contribution of our MO-ABC algorithm is the definition of specific local searches and the decomposition of the normalized objective space in three sectors in order to directly control the probabilistic selection, and hence the search of the algorithm. Experimental results show that our proposal is very promising because it outperforms other algorithms published in the literature.

As a future work, it would be interesting to study the effectiveness of each operator (cell-based local search, LA-based local search, etc.) independently. Furthermore, it would be a good challenge to study other multiobjective metaheuristics [17] and compare them with our MO-ABC algorithm.

## References

1. Agrawal, D., Zeng, Q.: Introduction to Wireless and Mobile Systems. Cengage Learning (2010)
2. Almeida-Luz, S.M., Vega-Rodríguez, M.A., Gómez-Pulido, J.A., Sánchez-Pérez, J.M.: Applying Differential Evolution to a realistic location area problem using SUMATRA. In: Proceedings of The Second International Conference on Advanced Engineering Computing and Applications in Sciences, pp. 170–175 (2008)
3. Almeida-Luz, S.M., Vega-Rodríguez, M.A., Gómez-Pulido, J.A., Sánchez-Pérez, J.M.: Solving a realistic location area problem using SUMATRA networks with the Scatter Search algorithm. In: Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications, pp. 689–694 (2009)
4. Berrocal-Plaza, V., Vega-Rodríguez, M.A., Sánchez-Pérez, J.M., Gómez-Pulido, J.A.: Applying the Strength Pareto Evolutionary Algorithm to solve the location areas planning problem in realistic networks. In: 14th International Conference on Computer Aided Systems Theory, pp. 176–178 (2013)
5. Berrocal-Plaza, V., Vega-Rodríguez, M.A., Sánchez-Pérez, J.M., Gómez-Pulido, J.A.: Solving the location areas scheme in realistic networks by using a multi-objective algorithm. In: Esparcia-Alcázar, A.I. (ed.) EvoApplications 2013. LNCS, vol. 7835, pp. 72–81. Springer, Heidelberg (2013)

6. Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V.: *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus (2006)
7. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6(2), 182–197 (2002)
8. Demestichas, P., Georgantas, N., Tzifa, E., Demesticha, V., Striki, M., Kilanioti, M., Theologou, M.E.: Computationally efficient algorithms for location area planning in future cellular systems. *Computer Communications* 23(13), 1263–1280 (2000)
9. Garg, V.: *Wireless Communications & Networking*, 1st edn. Morgan Kaufmann Publishers Inc., San Francisco (2007)
10. Gondim, P.R.L.: Genetic algorithms and the location area partitioning problem in cellular networks. In: *Proceedings of the IEEE 46th Vehicular Technology Conference on Mobile Technology for the Human Race*, vol. 3, pp. 1835–1838 (1996)
11. Jannink, J., Cui, Y.: Stanford university mobile activity traces (sumatra), <http://infolab.stanford.edu/sumatra> (accessed in 2013)
12. Karaboga, D., Basturk, B.: Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds.) *IFSA 2007. LNCS (LNAI)*, vol. 4529, pp. 789–798. Springer, Heidelberg (2007)
13. Kyamakya, K., Jobmann, K.: Location management in cellular networks: classification of the most important paradigms, realistic simulation framework, and relative performance analysis. *IEEE Transactions on Vehicular Technology* 54(2), 687–708 (2005)
14. Rubio-Largo, A., Vega-Rodríguez, M.A., Gómez-Pulido, J.A., Sánchez-Pérez, J.M.: A multiobjective approach based on artificial bee colony for the static routing and wavelength assignment problem. *Soft Comput.* 17(2), 199–211 (2013)
15. Subrata, R., Zomaya, A.Y.: Dynamic location management for mobile computing. *Telecommunication Systems* 22(1-4), 169–187 (2003)
16. Wong, V.W.S., Leung, V.C.: Location management for next-generation personal communications networks. *Netwrk. Mag. of Global Internetwkg.* 14(5), 18–24 (2000)
17. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation* 1(1), 32–49 (2011)
18. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., da Fonseca, V.: Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation* 7(2), 117–132 (2003)



# Yield Optimization Strategies for (DNA) Staged Tile Assembly Systems

Eugen Czeizler and Pekka Orponen

Department of Information and Computer Science  
Aalto University P.O. Box 15400, FI-00076 Aalto, Finland  
`{firstname.lastname}@aalto.fi`

**Abstract.** The staged Tile Assembly Model has been introduced by Demaine et al. 2008 as an enhancement of the previous tile self-assembly model of Winfree. In this framework, the assembly is allowed to be performed in parallel in different test-tubes, and the obtained products are stored and mixed in subsequent assembly stages. Using elegant combinatorial constructions, it has been shown that staged assembly systems possess remarkable advantage in comparison to their abstract counterparts. Because of their parallel nature, one can choose from a multitude of staged assembly strategies for assembling a given target structure. In the current work we analyze these assembly variations from a kinetic perspective, in order to determine and possibly maximize, their final assembly yield. As a pre-requirement for this task, we provide a procedure for associating an analytically tractable mathematical model to a given staged assembly experiment, based on which we can predict the yield concentration of the final assembly product. As a case study, we consider various assembly strategies as well as optimized and non-optimized assembly protocols for generating a size-10 tile assembly.

**Keywords:** Tile Assembly Model, staged assembly, numerical modelling, yield optimization.

## 1 Introduction

The abstract Tile Assembly Model (aTAM) has been introduced by Winfree [9] as a custom-made generalization of Wang tile systems, designed for the study of DNA tile self-assembly. The basic components of the aTAM are non-rotatable unit square tiles, uniquely defined by the sets of four glues placed on top of their edges. The glues are part of a finite alphabet and each pair of glues is associated a strength value, determining the stability of the link between two tiles having these glues on the abutting edges. The assembly process starts from a single nucleation point, the seed, and it continues by sequential attachments of tiles until no more tiles can be added to the assembly. All the individual tiles are placed inside a unique assembly “pot”, and the assembly process progresses with no external interactions.

In order to improve the efficiency of these systems, with respect to assembling more complex structures from a fewer initial number of distinct *tile-types*<sup>1</sup>, Demain et al. introduced the staged Tile Assembly Model (sTAM) [2]. In this framework, the assembly is performed in stages and in different *test-tubes* (or *bins*). Each test-tube is initialized with one or several non-interacting tile-types, and in each stage, one or several test-tubes are mixed together according to a predefined scheme. Different tile-types are thus mixed into the same compartment and start interacting. No seed structures are defined in this framework, and thus the reactions are implemented population-wise. The external observer allows the reaction to progress for some time, after which the content of the test-tubes is filtered and only the generated reaction products are used in subsequent stages.

Using elegant combinatorial designs, Demain et al. [2,3] demonstrated how various structures can be assembled efficiently, both in terms of the total number of different tile-types used, and in terms of the tile-interaction complexity, i.e., using only temperature-1 systems<sup>2</sup>. For example, one requires only 3 tile-types and  $\log(n)$  stages for constructing an  $n$ -size ribbon of contiguous tiles, while a similar structure assembled in a “one pot” system, i.e. classical aTAM, requires  $n$  distinct tile-types. Similarly, using a constant number of tile-types and only  $\log(n)$  stages, one can assemble a full  $n \times n$  square, whereas in the aTAM framework  $O(\log n / \log \log n)$  tile-types are required to assemble an analogous structure.

Because of the parallel design feature, one can choose from a multitude of staged assembly strategies for assembling a given target structure. Moreover, this freedom of choosing between several assembly variants remains valid even when one restricts to those strategies employing a minimum number of assembly stages. In the current work we analyze these assembly variations, as well as possible different implementations of the same assembly strategy, all in terms of their predicted final yield. Our objective is to study possible yield optimization protocols for the target assembly of these system. Considering assembly systems with an abundance of inter-molecular interaction (as is the case of DNA self-assembly systems), putting together larger concentrations of reactants and allowing them more time to react will always generate better yields. Thus, in order to perform a fair comparison between various assembly strategies, we require the total initial reactant concentrations, volume, as well as total time allowed for the reactions, to be constant in all of the compared strategies.

The particular aspects we want to investigate are:

- the yield variations in between different staged assembly schemes (generating the same final structures); and
- the yield variations within the same assembly scheme, when modifying parameters, such as: i) the time allocation for each of the assembly stages (while

---

<sup>1</sup> A tile-type is a population of identical copies of the same tile.

<sup>2</sup> Temperature-1 systems are highly advantageous as they can be made very resistant to errors, compared to temperature 2 systems [1]. Such systems can be implemented using e.g. DNA-origami techniques [8].

the total time for the experiment remains constant); or ii) the ratio in which certain assemblies are mixed inside the test-tubes (with total volumes of the reactants kept constant).

The first criterion can be seen as a design optimization level, while the second as a protocol optimization level. Moreover, we ask whether there exists a correlation between the two levels. Namely, would a design scheme performing particularly well on some assembly protocol generally generate better yields (than other assembly schemes) independent of the employed protocols?

In order to be able to address such questions, we first provide a methodology of assigning to every staged tile assembly system (sTAS) a numerical model describing the time-evolution of all its components. The employed modelling methodology is based on the principle of mass-action kinetics [5,6] and is implemented using the formulation given by ordinary differential equations (ODE). The modeling methodology is different from the one considered in the kTAM models, [12], as in this case we do not follow the assembly of only one particular structure (starting from a seed tile), but we keep track of all the species available in solution(s). While such an approach is usually untractable for “one pot” systems, we show that it becomes applicable in the case of sTAS. We use the above modelling methodology and, as a case study, we consider the assembly (and yield optimization process) of a size-10 tile assembly structure. For numerical modelling and optimization we have used the open source software COPASI [7].

The paper is organized as follows. The next section contain background information regarding the aTAM and sTAM models. In Section 3 we introduce our kinetic modelling methodology for sTAS and provide a series of pre-normalization requirements for our models. In Section 4 we introduce several yield optimization strategies applicable to staged assembly systems, and as a case study in the next section we consider the staged assembly and yield optimization protocols employed in obtaining size-10 horizontal ribbons of tiles. In the last section we discuss our results and provide some future research directions.

## 2 Background

In the following, we provide a very brief introduction of the (abstract) Tile Assembly Model, aTAM, and its staged counterpart, sTAM. For a more detailed presentation of these models we refer to [9,2] as well as the recent survey [4].

Let  $\Sigma$  be a finite set of *glues*, and let  $s : \Sigma \times \Sigma \rightarrow \mathbb{N}$  be a *glue strength* function, i.e.,  $s(\sigma_1, \sigma_2) = s(\sigma_2, \sigma_1)$  for all  $\sigma_1, \sigma_2 \in \Sigma$ . A *tile* (or *tile-type*)  $t$  is a unit square structure with glues on its four edges; we assume that the tiles can not be either rotated or reflected. Thus, we can represent a tile as the ordered 4-tuple of glues  $t = (t_N, t_E, t_S, t_W) \in \Sigma^4$  where the  $N, S, E,$  and  $W$  subscripts point to the corresponding edge positioning. An *assembly*  $\mathcal{A}$  is a partial mapping  $\mathcal{A} : \mathbb{Z}^2 \rightarrow \Sigma^4$  assigning tiles to locations in the two-dimensional grid, such that the defined structure is connected. A *tile assembly system* (TAS)  $\mathcal{T} = (T, \mathcal{S}, s, \tau)$  consists of a finite set  $T$  of tile-types, an assembly  $\mathcal{S}$  called the *seed assembly*,

a glue strength function  $s$  and a *temperature*  $\tau \in \mathbb{Z}^+$ . By definition, we assume that the seed assembly  $S$  is stable and cannot be disassembled<sup>3</sup>.

Given a TAS  $\mathcal{T} = (T, \mathcal{S}, s, \tau)$  and an assembly  $\mathcal{A}$  (such as the seed  $S$ ), a new tile can be added to  $\mathcal{A}$  if it shares a common boundary with tiles that bind it into place with total strength at least  $\tau$ ; we call such a process a *successful tile addition*. We say that a TAS  $\mathcal{T}$  *produces* an assembly  $\mathcal{A}$  if this assembly is formed by a sequence of successful tile additions starting from the seed assembly  $S$ . Moreover, if no other tiles can be further attached to  $\mathcal{A}$ , we say that the assembly is *terminal*.

The model of staged assembly differs from the classical aTAM by allowing partial assemblies to be formed in parallel in different test-tubes before merging them together. The notion of successful addition is extended from the previous case by allowing the merging of any two assemblies, as long as the sum of the strength of glues placed along the common boundary of the two assemblies is at least the temperature  $\tau$  of the system. Thus, in this setting, single tiles are seen just as (elementary) assemblies. The above requirement for an assembly is known as *partial connectivity*<sup>4</sup>, see [2], as it does not enforce tiles in the assembly to have matching edges with all the neighboring tiles, as long as the matching which bound them into place exceed or are equal with the temperature  $\tau$ . For the remaining of this paper we assume working in this partial connectivity requirement for assemblies.

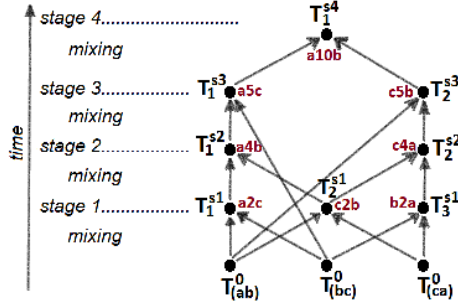
Another difference from aTAM comes from the fact that the assembly process is allowed to be performed in parallel in different *test-tubes* (or *bins*) and across several assembly stages. Each tile-type is placed initially in an isolated test-tube; we call these *initial test-tubes*. When the content of two (or several) test-tubes is mixed in a separate bin, the assemblies start interacting and bind to each-other according to their glue interactions. The process is allowed to progress for some time, after which the mixed solution is filtered and only the reaction products, i.e., the terminal assemblies, are stored for further mixing, while the remaining reactants are discarded. The test-tubes are further mixed synchronously during several assembly stages, until the final product is assembled in the unique test-tube of the last assembly stage.

A staged tile assembly system (sTAS)  $\mathcal{T}^s = (T, s, \tau, G)$  is defined by the set  $T$  of starting tile-types  $t_i$ , each placed in marked *initial test-tubes*  $T_{t_i}^0$ , a glue strength function  $s$ , a temperature parameter  $\tau$ , and an *assembly graph*  $G$  (or *mix graph*). The assembly graph is a direct acyclic graph (DAG) describing the different test-tubes and the way these tubes are mixed along a synchronous succession of assembly stages. The nodes of the graph are the various test-tubes (including the initial ones), while a directed edge between two nodes  $T^{s_i}$  and  $T^{s_j}$  symbolizes that the assembly product of test-tube  $T^{s_i}$  (or the corresponding tile-type in case of an initial test-tube) is transferred (either completely if  $T^{s_i}$  has no other out-edges or just a fraction of it otherwise) to test-tube  $T^{s_j}$ . The

---

<sup>3</sup> On some experimental implementations of the TAM, the seed assembly is implemented using e.g., DNA origami [10,11]

<sup>4</sup> As opposed to the *full connectivity* requirement.



**Fig. 1.** The annotated assembly graph of an sTAS assembling a size-10 ribbon

final assembly of the sTAS is the assembly product collected at the end of the experiment from the unique test-tube of the last assembly stage.

As an example, in the following we provide an sTAS assembling a size-10 horizontal ribbon of tiles. Since this is a 1D structure, only the glues of the East and West sides of a tile-type are relevant for the assembly process. Thus, a tile is denoted as the pair  $(x, y)$  of its West and East glues, respectively. Moreover, a 1D assembly containing  $k > 1$  tiles will be denoted as  $xky$ , where  $x$  (resp.  $y$ ) is the West (East) glue of the left-most (right-most) tile in the assembly. The temperature  $\tau$  of the system is 1, and the strength function  $s$  is given by  $s(x, y)$  is 1 if  $x = y$  and 0 otherwise. The sTAM contains 3 initial test-tubes  $T_{(ab)}^0, T_{(bc)}^0, T_{(ca)}^0$  for the tile-types  $(a, b)$ ,  $(b, c)$ , and  $(c, a)$ , respectively, and employs 4 assembling stages. The assembly-graph from Figure 1 fully describes the design of the sTAS; for ease of understanding we have annotated the graph by providing also the description of the assembly product in each of the test-tubes.

Various 2D assembly structures can be efficiently<sup>5</sup> assembled by appropriate staged assembly systems, even at temperature  $\tau = 1$  and using only two reactants per test-tube<sup>6</sup>, see e.g. [2]. Although the results of our current research apply to both 1D and 2D assembled structures, in order to simplify the considered mix graph designs and exemplify the applicability of our approach, are going to concentrate over the assembly of 1D ribbons of tiles. Indeed, if more complex 2D assemblies are investigated, the only change comes in the design of the mixing graph. However, the dynamics of the system is preserved, as mixing a size- $p$  assembly (i.e., containing  $p$  tiles) with a size- $q$  assembly, always generates a size- $(p + q)$  assembly, assuming the two components are indeed reacting.

Thus, from now on, we represent the tiles as the pairs of glues placed on their West and East edges, respectively, we assume working always at temperature 1, and we use the strength function given by  $s(x, y) = 1$  if  $x = y$  and 0 otherwise.

<sup>5</sup> Here, we measure the efficiency in terms of the number of different tile-types used

<sup>6</sup> In most of the staged assembly designs from the literature, only two reactants are placed inside a test-tube.

### 3 Modeling of Staged Tile Assembly Systems

In order to be able to address questions regarding yield optimization of sTAS we need appropriate quantitative tools for estimating and analyzing the corresponding yields. In this section we introduce an adequate mathematical model of the staged assembly process. Using this methodology, for any particular target structure, one can numerically determine the best assembly strategy for it, as well as numerically optimize the parameters of the chosen assembly strategy.

The modelling paradigm that we choose to use is that of ODE, while the formulation of the models is based on the principle of mass-action kinetics. The principle of mass-action, introduced in [5,6], says that the rate of each reaction is proportional to the concentration of reactants. Moreover, this reaction rate gives the measure on which the reactants are consumed and the products are generated. To exemplify, consider the simple reaction  $A + B \rightarrow A : B$  when an assembly  $A$  joins an assembly  $B$  and forms an assembly  $A : B$ . If we denote by  $[A](t)$ ,  $[B](t)$ , and  $[A : B](t)$  the concentrations these assemblies at time  $t$ , and by  $k$  the kinetic rate constant of the reaction, then the combined measure of consuming and producing each of the reactants is given by the system:

$$\frac{d[A]}{dt} = -k[A] \cdot [B] \quad \frac{d[B]}{dt} = -k[A] \cdot [B] \quad \frac{d[A : B]}{dt} = k[A] \cdot [B]$$

We are going to assume (without loss of generality, but with some possible loss of design efficiency) that in each stage of the assembly, we allow to mix the contents of only two test-tubes at a time; most of the sTAS in the literature are nevertheless designed in this way. Moreover, before the mixing procedure, the content of each test-tube is filtered and only the product of the assembly is preserved. Thus, in each test-tube we have only two reactants. As a consequence of this, the chemical reaction system corresponding to each of the test-tubes (each test-tube generates an isolated system) obeys two conservation reactions. Namely, at any time point  $t$  we have that

$$[A](t) + [A : B](t) = C_1 \text{ and } [B](t) + [A : B](t) = C_2,$$

for two constants  $C_1$  and  $C_2$ , such that  $C_1 = [A](0) + [A : B](0)$  and  $C_2 = [B](0) + [A : B](0)$ <sup>7</sup>. Thus, at any time point  $t$ , the concentration of the  $[A]$  and  $[B]$  species can be derived from the concentration of the  $[A : B]$  species. By substituting these into the third differential equation we obtain:

$$\frac{d[A : B]}{dt} = k(C_1 - [A : B])(C_2 - [A : B]) \quad (1)$$

In most cases, such ODE systems derived from corresponding chemical reaction systems are analytically intractable. However, since in the case of sTAS we have that in each test-tube there exist only two reactants interacting and forming a

---

<sup>7</sup> We denoted by  $[A](0)$ ,  $[B](0)$ , and  $[A : B](0)$  the initial concentration of the species  $A$ ,  $B$ , and  $A : B$ , respectively, at time  $t_0 = 0$ .

product (a larger complex), the derived ODE systems can be solved analytically. Namely, equation (1) has the solution:

$$[A:B](t) = \frac{-C_1 C_2 + [A:B](0)C_1 + C_1 C_2 e^{tk(C_1-C_2)} - [A:B](0)e^{tk(C_1-C_2)}}{C_1 e^{tk(C_1-C_2)} - [A:B](0)e^{tk(C_1-C_2)} - C_2 + [A:B](0)} \quad (2)$$

Another particularity of sTAS is that none of the  $[A:B]$  structures exist before mixing assemblies  $A$  and  $B$ , that is  $[A:B](0) = 0$ . Thus, equation (2) becomes:

$$[A:B]_t = \frac{-C_1 C_2 + C_1 C_2 e^{tk(C_1-C_2)}}{C_1 e^{tk(C_1-C_2)} - C_2} = \frac{C_1 C_2 (e^{tk(C_1-C_2)} - 1)}{C_1 e^{tk(C_1-C_2)} - C_2}, \quad (3)$$

where  $C_1 = [A](0)$  and  $C_2 = [B](0)$ . Moreover, if one also assumes that  $[A](0) = [B](0) = C$ , i.e., the systems is symmetric, then equation (2) becomes

$$[A:B]_t = \frac{ktC^2}{1 + Ckt}. \quad (4)$$

Because at each stage of the assembly the initial concentrations for the reactants depend on the concentrations of the products at prior stages, and since equation (1) describing the time-evolution of the product assembly in each test-tube admits an analytic solution, we can provide an analytic formula for the entire system.

An important observation regarding the dynamics of sTAS is that the products obtained in prior stages of the assembly are not further concentrated before mixing them in subsequent stages. Thus, in each stage, the volume of the solution increases, and hence we have to update the concentration of the reactants accordingly (i.e., to decrease these concentrations).

For example, assume the reactants  $R_1$  and  $R_2$  of test-tube  $T$  from some stage of the assembly are taken to be fractions of the products  $P_1$  and  $P_2$  of test-tubes  $T1$  and  $T2$ , respectively (from some previous stages). Namely, let

$$Vol_{trans}^{P_1} = r_{T1}^T \cdot Vol_{T1} \quad \text{and} \quad Vol_{trans}^{P_2} = r_{T2}^T \cdot Vol_{T2}$$

be the volumes of the fraction of products  $P_1$  and  $P_2$  transferred from  $T1$  and  $T2$  respectively, to  $T$ , where  $Vol_{T1}$  (resp.  $Vol_{T2}$ ) and  $r_{T1}^T$  (resp.  $r_{T2}^T$ ) denote the volume of test-tube  $T1$  (resp.  $T2$ ) and the ratio from this volume which is transferred into  $T$ . Then, the initial concentration of reactants  $R_1$  and  $R_2$  in test-tube  $T$  is given by

$$[R_1](0) = \frac{[P_1] \cdot Vol_{trans}^{P_1}}{Vol_{trans}^{P_1} + Vol_{trans}^{P_2}}; \quad \text{and} \quad [R_2](0) = \frac{[P_2] \cdot Vol_{trans}^{P_2}}{Vol_{trans}^{P_1} + Vol_{trans}^{P_2}}, \quad (5)$$

where  $[P_1]$  (resp.  $[P_2]$ ) is the concentration of the product  $P_1$  (resp.  $P_2$ ) at the end of the corresponding stage, and  $Vol_{trans}^{P_1} + Vol_{trans}^{P_2} = Vol_T$  is the volume of the test-tube  $T$ .

Thus, by keeping track of the volumes of each test-tube and knowing the ratio in which a particular product is split, we can determine the analytic formula of each intermediary (or final) species in the system.

Once such a computational model is derived, it can be estimated numerically for various sets of parameters, e.g., equal time-splits for all stages and/or equal (or proportional) volume-splits of various products. Moreover, the above parameters can be optimized in order to maximize the yield (i.e., concentration) of the final product. Also, using such models, we can compare two or several strategies in determining which provides a better yield, if experiments are performed in similar conditions, i.e., same total time and initial tile concentration.

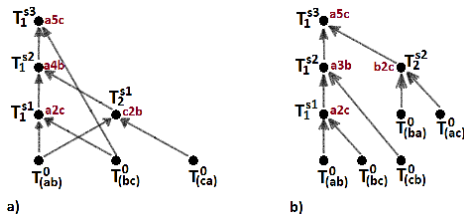
In order to compare two (or several) assembly strategies we can further simplify the models by making a synchronous pre-normalization of the data. Thus, we are going to assume from now on that the kinetic rate constant of all assembly reactions is equal to 1, and that the concentration of all tile-types in their initial test-tubes,  $[mon]$ , is also normalized to  $[mon] = 100$ . Because of the above pre-normalization of the data, the time parameter presents a highly altered behaviour; thus, from now on, we use the notion of *time unit* (*t.u.*) for referring to time variables. Consider for example a system of only two reactants (tile-types)  $a1b$  and  $b1c$ , each having concentration 100 in their initial test-tubes. Assuming these reactants are mixed in equal quantities, their initial concentration in the (mixing) test-tube becomes 50. In these conditions, we observe that the assembly reaction is completed in proportions of approx. 50%, 75%, and 90% only after 0.02, 0.06, and 0.2 t.u., respectively. Thus, our in-silico experiments and numerical analysis will be performed for a total time interval of 0.14–0.25 t.u. per stage, that is 0.42 t.u. for the 3-staged assembly of size-5 ribbons (in Section 4), and 0.9 t.u. for the 4-staged assembly of size-10 ribbons (in Section 5).

## 4 Yield Optimization Strategies for sTAM

**Optimizations at the Assembly Strategy Level.** The sTAM framework allows for several assembly strategies to be employed in achieving the same final structure. Moreover, in some cases, each of these strategies, although different in themselves, are all optimal in terms of number of distinct test-tubes or stages they employ. Consider for example the staged assembly process needed for assembling a size-5 ribbon. According to the assembly designs introduced in [2] for constructing size- $k$  1D ribbons of tiles in optimally possible number of stages, there are four different (staged) assembly strategies for the construction of size-5 ribbons, each employing 3 stages. The assembly graphs of two of these strategies are provided in Figure 2, while the remaining two strategies are symmetric. However, does each of these strategies produce the same amount of final yield (assuming that the initial quantity of resources is proportionally equal in each of the situations)?

In order to compare the previous two strategies, besides the common total time for the experiment ( $T_{total} = t_1 + t_2 + t_3 = 0.42$  t.u.) and the similar concentration ( $[mon] = 100$ ) of all tile-types in their initial test-tubes, in both scenarios we are





**Fig. 2.** Two distinct assembly strategies for the same size-5 ribbon, each using only 3 assembling stages

going to use a similar procedure of setting the time- and volume-split parameters, as follows:

- All assembly stages are performed in equal time intervals:  $t_1 = t_2 = t_3 = 0.14$ ;
- Whenever a tile-type (monomer) is a reactant in a test-tube, the introduced quantity of this reactant in the test-tube is exactly one unit volume.

By numerically estimating the associated mathematical models we obtain the concentration of the final product,  $[a5c]$ , in the assembly scenarios from Figure 2 a) and b) as 54.4% and 56.2% respectively, where 100% would represent the all-maximal value possible for this assembly, e.g. obtained if time would allow the reactions to be fully completed<sup>8</sup>.

From the above example, it can be confirmed our initial assumption that different assembly strategies may generate different final yields, despite using the same amount of time and substance resources.

**Optimizations at the Experimental Setting Level.** Consider now we have chosen a particular assembly strategy, say e.g., assembling the previous 5-tile structure by the scenario in Figure 2 a). A subsequent question concerns the way of allocating the total pool of resources, i.e., substance volume in each test-tube, time allocation for each of the assembly stages, etc., such as to maximize the outcome of the experiment. For example, in the case of the previous example, what would be the best split of the total time of the experiment into three time-periods for the corresponding assembly stages, such as to obtain a maximum amount of 5-tile structures at the end of the final stage? Also, what would be the best way of splitting the amount of tile  $(bc)$  in between  $T_1^{s1}$  and  $T_1^{s3}$ , or similarly the splitting of the amount of tile  $(ab)$ ? Also, for the cases when an intermediary assembly is used in several reactions from some later stages, what is the optimal way of splitting this product, i.e., its total volume, in between these test-tubes? The above time- and volume-splitting ratios can be subjected to targeted optimization protocols.

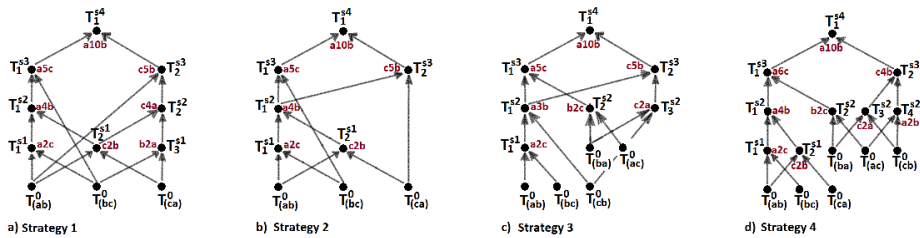
<sup>8</sup> In the case of 5-tile ribbons, 100% corresponds to a concentration of 20.

Until now, as suggested by the definition of the sTAM, we have assumed that the entire volume of an intermediate product is transferred to the subsequent stages. However, as suggested by actual lab procedures, we examine also the setting in which only a fraction of the reaction products are re-introduced as reactants. Namely, we force that in every test-tube, the volume of the two reactants sum up to exactly one. In this setting, it becomes even more clear that the ratio in which the two reactants are mixed (each coming into the reaction with possible different initial concentrations) becomes very important in determining a maximum concentration of the product.

## 5 Case Study: Assembling Size-10 Ribbons

As an yield optimization case study, we are going to consider the process of assembling (in a staged assembly fashion) a size-10 1D horizontal ribbon of tiles. As previously explained, restricting to this 1D structure is not a considerable limitation, since even in the case of assembling 2D complexes, once the mixing graph is designed, the modeling and optimization procedures remain the same.

Since the available pool of possible assembly strategies for a size-10 horizontal line is considerably large, even for the case where we impose using only four stages, we are going to compare only four particular such strategies<sup>9</sup>. We present these strategies (a.k.a. the corresponding mixing graphs) in Figure 3.



**Fig. 3.** Four distinct assembly strategies for a size-10 ribbon, each using only 4 assembling stages

In order to illustrate the possible differences between different assembly scenarios as well as between optimized vs. non-optimized assembly protocols, we do not restrict to computing only the optimum values, but provide for comparison a larger pool of parameter setups. Thus, we are comparing all these four assembly strategies, by subjecting each of them to five different setups regarding their time-split and volume-split parameters. Three of these setups are based on combinatorial heuristics, while in two of them we numerically optimize the parameters for maximizing the final yield concentration. In order to compare all of these strategies and setups, we impose some general constraints as follows:

<sup>9</sup> These strategies have been chosen almost at random from the available ones, without a prior knowledge on their behaviour during the optimization process.

- i) The total time of the experiment is  $T_{total} = 0.9$  t.u.
- ii) The concentration of all tile-types in their initial test-tubes is  $[mon] = 100$ .
- iii) (only for three of the setups) The cumulative volume of all tile-types introduced in the various initial test-tubes is 10 unit volumes, where the volume of each tile-type is proportional to the number of times it appears in the final size-10 assembly.

The five setups can be partitioned into two groups:

### Group 1: Combinatorially designed setups:

*Setup 1: Equal time-splits and proportional volume-splits.*

- Equal time intervals for the stages, that is,  $t_1 = t_2 = t_3 = t_4 = 0.225$ ;
- The volume of any species who needs to be partitioned into two or several test-tubes will be done so proportionally to how much the product of these latter test-tubes will contribute to the final assembly.

A combinatorial observation regarding staged assembly systems is that on average the concentration of the reactants is reduced at least by half in each stage. By inspecting equation (3), we observe that if the concentration of both reactants is reduced by half, we obtain the same product-reactant ratio only if we double the time allocated to this stage. Thus, as a possible procedure for improving the overall yield, the time-split parameters from the next setup are in geometric progression.

*Setup 2: Time-splits in geometric progression and proportional volume-splits.*

- The time intervals are  $(t_1; t_2; t_3; t_4) = (0.06; 0.12; 0.24; 0.48)$ ;
- The volume of products is partitioned proportionally into subsequent test-tubes (as in the case of Setup 1).

*Setup 3: Equal time-splits and equal half unit-volumes for all reactants*

- Equal time intervals:  $t_1 = t_2 = t_3 = t_4 = 0.225$ ;
- The volume of each of the reactants in a test-tube is set to half unit volume.

This represents a rather “lazy” (or automated) instance of the setting in which the volume of the test-tubes is limited to one unit.

### Group 2: Numerically optimized setups:

*Setup 4: Optimized time- and volume-splits while using the entire volume of substance.* As in the case of Setup 1 and 2, we assume here that we completely use the entire volumes of all intermediary assemblies and of all single tiles.

**Table 1.** The concentration [a10b] of the size-10 ribbon structure generated in the final stage of the assembly; the results are expressed in their percentage form, where 100% represents the absolute maximal value possible for this assembly, namely 10

Assembly strategy	Group 1			Group 2	
	Setup 1	Setup 2	Setup 3	Setup 4	Setup 5
Strategy 1	41.4%	34.7%	25.0%	44.5%	53.5%
Strategy 2	41.4%	34.7%	25.0%	44.5%	53.5%
Strategy 3	42.6%	42.0%	30.5%	46.9%	49.0%
Strategy 4	42.4%	39.8%	37.3%	49.3%	48.6%

*Setup 5: Optimized time- and volume-splits while enforcing unit volumes for all test-tubes* As in the case of Setup 3, we enforce that each test-tube contains exactly one unit of mixed reactants.

All four assembly scenarios are subjected to the above setups, and the results are summarized in Table 1. As it can be seen from the selected assembly scenarios, in most cases the differences are relative small. However, there exist both particularly bad and particularly good cases. Namely, the average value of the produced yield is 41.3%, the sample standard deviation is 8.2, the worst case scenario gives a yield percentage of 25%, while the best case scenario provides a yield percentage of 53.5%. It is very interesting to observe that both the best and the worst case scenarios are due to the same assembly strategy, but from different, i.e., non-optimized vs. optimized, parameter setups. Also, it can be observed that for each of the setups, the yield percentage are closed from one assembly scenario to the other, thus suggesting a possible ranking of how good each of these individual setups are. Namely, we are able to say that the worst parameter setting is performed in Setup 3 while if instead of just placing the previous default values we numerically optimize them to maximize the yield, i.e., Setup 5, then we obtain the best possible results from all the considered parameter setups.

## 6 Conclusions, Discussions, and Further Work

We have investigated yield optimization techniques for staged self-assembly systems. As a first step, we associated (for the first time) a computational model to the staged tile assembly formalism, whose implementation through ODE systems differs considerably from the kinetic counterpart of the regular TAM. This change of modelling methodology can be explained as follows. While in case of abstract TAM the assembly is initiated from a seed structure, and one can thus concentrate over a single assembly product, in case of sTAM the assembly reactions are implemented population wise.

Another important aspect was to determine the possible optimization strategies for our target, the final assembly yield. We were able to identify two levels on which to implement adequate optimization protocols: at the assembly scheme

design level, and at the implementation level. Considering the first level, several assembly strategies are plausible for the same final structure, and some of these assembly schemes may have plausible better chances of maximizing the concentration of the final product. We concentrate here only on those assembly schemes which ensure a minimal number of assembly stages. We conjecture at this level that the best assembly protocols are those in which we minimize the number of mixing of test-tubes from the same stages. The intuition here is that the more advance a stage is, the less concentrated its product, and thus by mixing a test-tube with another one from a lower stage, the concentration of the latter is higher and thus it improves the result of the reaction.

The second optimization level is at the implementation phase, once a particular assembly strategy has been chosen. At this level, the parameters which can be optimized are the time intervals allocated to each of the assembly stages (assuming the total available time is constant) and the proportions in which certain products are split and further mixed in subsequent stages. We believe that equal time-splits are not an optimal choice (unless the experiment involves a low number of stages), but the considered case-study showed that time-splits in geometric progression are also not appropriate (i.e., Setup 2 in Table 1). Although the case study seems to indicate that the optimal time-split parameters are close to an arithmetic progression (Setup 4 and 5 in Table 1, data on time-splits not shown), we believe that further studies are required for providing more intuition regarding a possible combinatorial design approaching the absolute optimum choice.

Regarding volume splits, an indiscriminating equal partitioning seem to be the worst possible choice (Setup 3 in Table 1). On the opposite direction, those partitions of assembly products which take into considerations the amount (or concentration) of substance in each test-tube, and how the product of these test-tubes are further going to be split, tend to have better yields.

Considering the case-study, one particular assembly strategy has generated good outcomes, namely that of requiring the volume of each test-tube to be exactly one. For the future, we plan to concentrate particularly on this strategy, both because it seem to provide the best results, and because it seem to be more tractable from an analytic point of view. Our aims are to provide concrete descriptions of combinatorial parameter-setups and mix-graph designs for which we could provide numeric arguments as why the results of these strategies approach the optimum solutions.

## References

1. Cook, M., Fu, Y., Schweller, R.: Temperature 1 self-assembly: deterministic assembly in 3d and probabilistic assembly in 2d. In: Proc. of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, pp. 570–589. SIAM (2011)
2. Demaine, E.D., et al.: Staged self-assembly: nanomanufacture of arbitrary shapes with  $o(1)$  glues. *Natural Computing* 7(3), 347–370 (2008)
3. Demaine, E.D., Eisenstat, S., Ishaque, M., Winslow, A.: One-dimensional staged self-assembly. *Natural Computing* 12(2), 247–258 (2013)

4. Doty, D.: Theory of algorithmic self-assembly. *Commun. ACM* 55(12), 78–88 (2012)
5. Guldberg, C.M., Waage, P.: Studies concerning affinity. *C. M. Forhandlinger: Videnskabs-Selskabet i Christiana* (Norwegian Academy of Science and Letters) 35 (1864)
6. Guldberg, C., Waage, P.: Concerning chemical affinity. *Erdmanns Journal fr Practische Chemie* 127 (1879)
7. Hoops, S., et al.: Copasi - a complex pathway simulator. *Bioinformatics* 22(24), 3067–3074 (2006)
8. Padilla, J.E., Patitz, M.J., Pena, R., Schweller, R.T., Seeman, N.C., Sheline, R., Summers, S.M., Zhong, X.: Asynchronous signal passing for tile self-assembly: Fuel efficient computation and efficient assembly of shapes. In: Mauri, G., Dennunzio, A., Manzoni, L., Porreca, A.E. (eds.) *UCNC 2013. LNCS*, vol. 7956, pp. 174–185. Springer, Heidelberg (2013)
9. Rothmund, P.W.K., Winfree, E.: The program-size complexity of self-assembled squares. In: *Proc. 32nd Annual ACM Symposium on Theory of Computing (STOC 2000)*, pp. 459–468. ACM, New York (2000)
10. Schulman, R., Winfree, E.: Synthesis of crystals with a programmable kinetic barrier to nucleation. *Proceedings of the National Academy of Sciences* 104(39), 15236–15241 (2007)
11. Schulman, R., Yurke, B., Winfree, E.: Robust self-replication of combinatorial information via crystal growth and scission. *Proceedings of the National Academy of Sciences* 109(17), 6405–6410 (2012)
12. Winfree, E.: Simulations of computing by self-assembly. Technical Report CaltechCSTR:1998.22, California Institute of Technology (1998)

# Online Gait Learning for Modular Robots with Arbitrary Shapes and Sizes

Massimiliano D'Angelo<sup>1</sup>, Berend Weel<sup>2,3</sup>, and A.E. Eiben<sup>3</sup>

<sup>1</sup> University La Sapienza, Rome, Italy

<sup>2</sup> University of York, UK

<sup>3</sup> VU University Amsterdam, The Netherlands

maxxi.d.angelo@gmail.com, {b.weel,a.e.eiben}@vu.nl

**Abstract.** This paper addresses a principal problem of *in vivo* evolution of modular multi-cellular robots. To evolve robot morphologies and controllers in real-space and real-time we need a generic learning mechanism that enables arbitrary modular shapes to obtain a suitable gait quickly after ‘birth’. In this study we investigate a reinforcement learning method and conduct simulation experiments using robot morphologies with different size and complexity. The experiments give insights into the online dynamics of gait learning, the distribution of lucky / unlucky runs and their dependence on the size and complexity of the modular robotic organisms.

**Keywords:** embodied artificial evolution, modular robots, artificial life, online gait learning, reinforcement learning.

## 1 Introduction

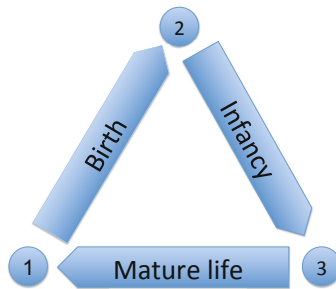
The work described in this paper forms a stepping stone towards the grand vision of embodied artificial evolution (EAE) as outlined in [6]. The essence of this vision is to construct physical systems that undergo evolution ‘in the wild’, i.e. not in a virtual world inside a computer. There are various possible approaches towards this goal including chemical and biological ones. The one behind this study is based on using a mechatronical substrate, that is, robots.

In general, there are two principal forces behind evolution: selection and reproduction. Selection –at least environmental, objective-free selection– is ‘for free’ in the real world. Therefore, the main challenge for EAE is reproduction, i.e., the creation of tangible physical artifacts with the ability to reproduce. In our case, this means the need for self-reproducing robots. The approach we follow to this end is based on modular robotics with robotic building blocks capable of autonomous locomotion and aggregation into complex ‘multicellular’ structures in 3D. This approach has several advantages. Firstly, it offers a high level of control because the basic modules or cells are robots themselves that can be predesigned and preprogrammed according to the preferences of the experimenter. Here one can choose a homogeneous system with identical building blocks or a heterogeneous one with several kinds of modules. Secondly, modular

robotics offers a high level of flexibility, because the number of different combinations, that is, different aggregated 3D structures, is huge. This means that the design space of all possible aggregated robotic organisms is rich enough to accommodate interesting evolutionary processes.

It is important to note that the morphology of the basic robot modules is fixed by design and cannot change during the operational / experimental period. Hence, evolution will not take place in the morphological space of these pre-engineered modules, but in the morphological space of the multicellular organisms. From the perspective of the multicellular robot bodies the basic robots are merely raw material whose physical properties do not change over time.<sup>1</sup>

Recently these ideas have been put on a more solid footing by presenting a conceptual framework for systems where robot morphologies and controllers can evolve in real-time and real-space [5]. This framework, dubbed the Triangle of Life, describes a life cycle that does not run from birth to death, but from conception (being conceived) to conception (conceiving one or more children) and it is repeated over and over again, thus creating consecutive generations of 'robot children'. The Triangle of Life consists of 3 stages, Birth, Infancy, and Mature Life.



**Fig. 1.** The Triangle of Life. The pivotal moments that span the triangle and separate the 3 stages are: 1) Conception: A new genome is activated, construction of a new organism starts. 2) Delivery: Construction of the new organism is completed. 3) Fertility: The organism becomes ready to conceive offspring.

In this paper we address a fundamental problem in the Infancy stage. This stage starts when the morphogenesis of a new robot organism is completed and the 'baby robot' is delivered. As explained in [5], the body (morphological structure) and the mind (controller) of such a new organism will unlikely fit each other well. Even if the parents had well matching bodies and minds, recombination and mutation can easily result in a child where this is not the case. Hence, the new organism needs some fine tuning; not unlike a newborn calf the 'baby robot' needs to learn how to control its own body. This problem –the Control

<sup>1</sup> Nevertheless, evolving the controllers of these elementary robot modules during the operational period is possible.



Your Own Body (CYOB) problem— is inherent to evolutionary ALife systems where both bodies and minds undergo changes during reproduction.

The work described here addresses the general CYOB problem in a simplified form, by reducing it to gait learning. In the modular robots approach the challenge is to find a method that can learn gaits for all different morphologies that can be created with the given modules and can do this quickly. The problem is highly nontrivial, since a modular robot organism has many degrees of freedom, which leads to a very large search space of possible gaits. Furthermore, this learning process must take place on-the-fly, during the real operational period of the robot organisms. The off-line approach, where a good controller is developed (evolved, learned, hand-coded, ...) before the robot is deployed is not applicable here, because the life cycle of the Triangle is running in a hands-free mode without being paused for intervention by the experimenter.

The mechanism we employ here to solve the CYOB problem is reinforcement learning, in particular, the PoWER algorithm described by Kober and Peters [14]. Note, that this learning mechanism is not evolutionary itself. Evolution takes place on the level of multicellular robot organisms (as it is these organisms that reproduce and get selected in the Triangle of Life framework), whereas the PoWER algorithm is applied inside one organism to discover a good controller that induces a good gait. The grand evolutionary process is not investigated here; it only forms the background context that raises the CYOB problem. The specific research questions our experiments will try to answer are the following:

1. How is the run-time dynamics of the learning process? That is, how quickly can a multi-cellular robot organism learn to walk?
2. How reliable is the learning process? That is, how often do we see lucky (unlucky) runs resulting in fast walking (immobilized) robot organisms?
3. How do the above features depend on the size and complexity of the robot morphologies?

## 2 Related Work

The design of locomotion for modular robotics is a difficult problem. As explained by Spröwitz: Locomotion requires the creation of rhythmic patterns which satisfy multiple constraints: generating forward motion, without falling over, with low energy, possibly coping with different environments, hardware failures, changes in the environment and/or of the organism [18]. In the literature there are several approaches, based on various types of controllers and algorithms for creating these rhythmic patterns.

One of the earliest types is gait control tables as in, for instance, [1] and [19]. A gait control table consist of rows of actuator commands with one column for each actuator, each row also has a condition for the transition to the next row, in essence this implements a very simple cyclic finite state machine. A second major avenue of research is that of neural networks (NN). In particular for locomotion of robot organisms HyperNEAT is used extensively. HyperNEAT is an indirect encoding for a neural network. The genome is a compositional pattern

producing network (CPPN), these networks are directed graphs in which each node is a mathematical function like sine, cosine or a Gaussian [4]. CPPN’s are used to set the weights of a fixed size neural network called a substrate. Several studies have shown that HyperNEAT is capable of creating efficient gaits for robots [4,20,9]. Another successful approach that has received much attention is based on Central Pattern Generators (CPG). CPGs model neural circuitry found in vertebrates which output cyclic patterns without requiring a cyclic input [11]. Each actuator in a robot organism is controlled by the output of a CPG, furthermore the CPGs are connected through certain variables which allows them to synchronise and maintain a certain phase difference pattern. Although sensory input is not strictly needed for CPG’s, it can be incorporated to shape the locomotion pattern to allow for turning and modulating the speed. This technique has been shown to produce well performing and stable gaits on both non-modular robots [18,2] and modular multi-robot organisms [13,12]. Last, a technique based on artificial hormones has been investigated for the locomotion of modular robot organisms. In this technique artificial hormones are created within robot modules as a response to sensory inputs. These hormones can interact with each other, diffuse to neighbouring modules and act upon output hormones. These output hormones are then used to drive the actuators [10,17]. Furthermore, some techniques in the field of gait learning employ reinforcement learning algorithms, the specific approaches used can range from Temporal Difference Learning (TDL) to Expectation-Maximization (EM). In TDL one seeks to minimize an error function between estimated and empirical results of a controller, in EM controller parameters are estimated in order to maximize the reward gained using it. These algorithms have been used on modular, e.g. [3] and non modular robots, e.g. [16].

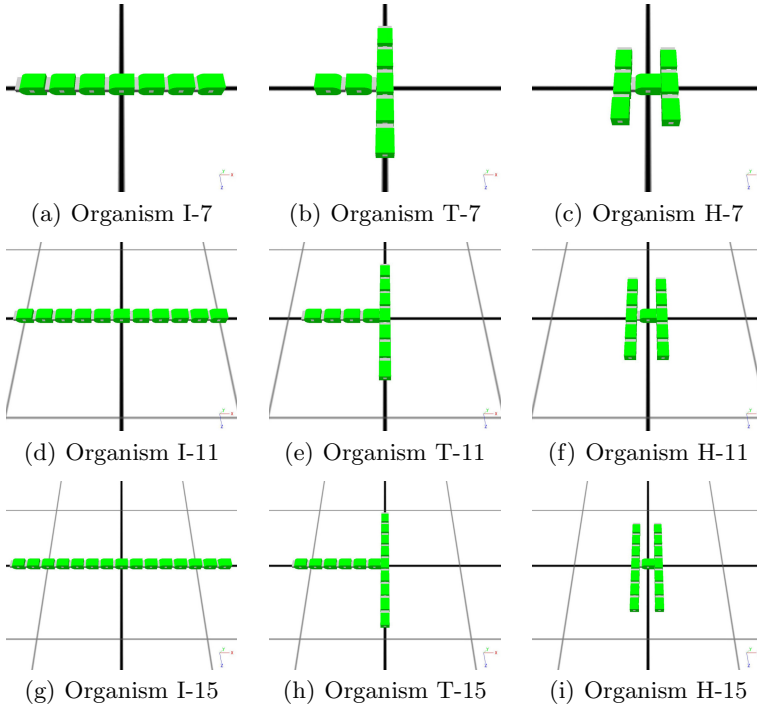
Although there is extensive previous work on this issue, we must stress that, of the techniques described above, only the techniques described in [3], [12] and [18] were actually tested on multiple shapes.

### 3 Experimental Setup

As mentioned in Section 1 our primary goal is to determine if a reinforcement learning (RL) approach can be suitable for online gait learning. To this end, we tested an RL algorithm in various organism morphologies (with different sizes and complexity) set in a simple environment. All these tests were done in simulation with the Webots system of Cyberbotics, using the YaMoR module as building block for the organisms [15]. A YaMoR module is made of a static body and a joint on its front that has a single degree of freedom and an operating range of  $[-\frac{\pi}{2}, +\frac{\pi}{2}]$ . It also has two connectors, one on the joint and one in the back of the body, which allow to connect modules at arbitrary angles. For the current investigation we applied three changes to the original YaMoR model. We added two extra connectors on the remaining sides of the body in a central position, allowing the construction of complex structures. We reduced the width of the joint to avoid its lateral protrusion, thereby eliminating the possibility of

collisions with the modules connected to the sides. Last, we added a GPS device to a centrally located module to measure displacement.

Nine different robot organisms with different sizes and complexity were defined so as to examine the algorithm generality and scalability. Size and complexity are measured by the number of modules and by the number of extremities, respectively. The experiments were conducted with three complexity levels: organisms with two extremities (I-shape), three extremities (T-shape), and four extremities (H-shape). Each shape was then replicated in three sizes: 7, 11 and 15 modules. A screenshot of their initial state can be seen in Figure 2.



**Fig. 2.** Robot Organisms

The environment chosen for the experiments is an infinite plane free of obstacles so to avoid any extra complexity and the need of supervision. Each experiment starts with the organism lying completely flat at the plane origin.

The controller of each organism defines an open-loop gait. Because we use reinforcement learning to acquire a good gait, we refer to our controllers as policies. In our implementation, a policy is a set of cyclic spline functions where each spline specifies the angular positions of an actuator over time. A cyclic spline is a mathematical function that can be defined using a set of  $n$  control

points. Each control point is defined by  $(t_i, \alpha_i)$  where  $t_i$  represents time and  $\alpha_i$  the corresponding value.  $t_i \in [0, 1]$  is defined as

$$t_i = \frac{i}{n-1}, \forall i = 0, \dots, (n-1) \quad (1)$$

and  $\alpha_i \in [0, 1]$  is freely defined, except that the last value is enforced to be equal to the first, i.e.  $\alpha_0 = \alpha_n$ . These control points are then used for cyclic spline interpolation using GSL [7] dedicated C functions. Using GSL it is possible to query a spline for a different number of points than it was defined with, enabling comparison between splines defined with a different number of parameters.

The problem is then to find a set of splines, that maximizes some measure of performance. For this we use the RL algorithm called PoWER described by Kober and Peters [14]. The use of the set of cyclic spline functions as the representation was taken from [16]. This RL algorithm is based on an Expectation-Maximization approach to estimate the parameters  $\hat{\alpha}$  of a policy  $\pi$  in order to maximize the reward gained by using that policy. The algorithm starts by creating the initial policy  $\pi_0$  with as many splines as there are robots (actuators). The algorithm initialises these splines with  $n$  values of 0.5 and then adding Gaussian noise. This initial policy is then evaluated after which it is adapted. This adapted controller is evaluated and adapted again until the stopping condition is reached.

Adaptation is done in two steps which are always applied: Exploitation and Exploration. In the exploitation step, the current splines  $\hat{\alpha}$  are optimized based on the outcome of previous controllers, this generates a new set of splines.

$$\hat{\alpha}_{i+1} = \hat{\alpha}_i + \frac{\sum_{j=1}^k \hat{\Delta}\alpha_{i,j} R_j}{\sum_{j=1}^k R_j} \quad (2)$$

where  $\hat{\Delta}\alpha_{i,j}$  represents the difference between the parameters of the  $i$ -th policy and  $j$ -th policy belonging to a ranking of the best  $k$  policies seen so far and  $R_j$  its reward. In the exploration phase policies are adapted by applying Gaussian perturbation to the newly generated policy.

$$\hat{\alpha}'_{i+1} = \hat{\alpha}_{i+1} + \hat{\varepsilon}_{i+1}, \hat{\varepsilon}_{i+1} \sim \mathcal{N}(0, \sigma^2) \quad (3)$$

where  $\hat{\alpha}_{i+1}$  are the parameters after the exploitation step,  $\hat{\alpha}'_{i+1}$  the parameters after the exploration step and  $\hat{\varepsilon}_{i+1}$  values drawn from a Gaussian distribution with mean 0 and variance  $\sigma^2$ . Over the course of evaluations, the variance  $\sigma^2$  is diminished which decreases exploration and increases exploitation.

To carry out the actual evaluation each policy is queried for its parameters at a rate proportional to the actuators’ angular speed. These are then appropriately scaled depending on actuators’ operating range and used cyclically during the evaluation. Each controller is evaluated for 96 seconds (6,000 time steps) after being used for a recovery period of 3.2 seconds (200 time steps) in order to reduce evaluation noisiness as in [8]. This is a rather long evaluation time for locomotion, but was chosen to be conservative and allow for a reasonably

**Table 1.** Experiment Parameters

Parameter	Value	Description
Variance	0.008	The initial variance
Variance Decay	0.98	The variance decay factor
Ranking Size	10	Number of best policies to compare against
Start Parameters	2	Starting number of parameters of a spline
End Parameters	100	Ending number of parameters of a spline
Evaluation Steps	6,000	Number of time steps for evaluations
Recovery Steps	200	Number of time steps for recovery
Evaluations	1,000	Number of evaluations

accurate measurement of performance. The reward  $R$  awarded to a controller  $i$  is calculated as follows:

$$R_i = \left( 100 \frac{\sqrt{\Delta_x^2 + \Delta_y^2}}{\Delta_t} \right)^6 \quad (4)$$

where  $\Delta_x$  and  $\Delta_y$  is the displacement over the  $x$  and  $y$  axes measured in meters and  $\Delta_t$  the time spent in evaluation, as in [16].

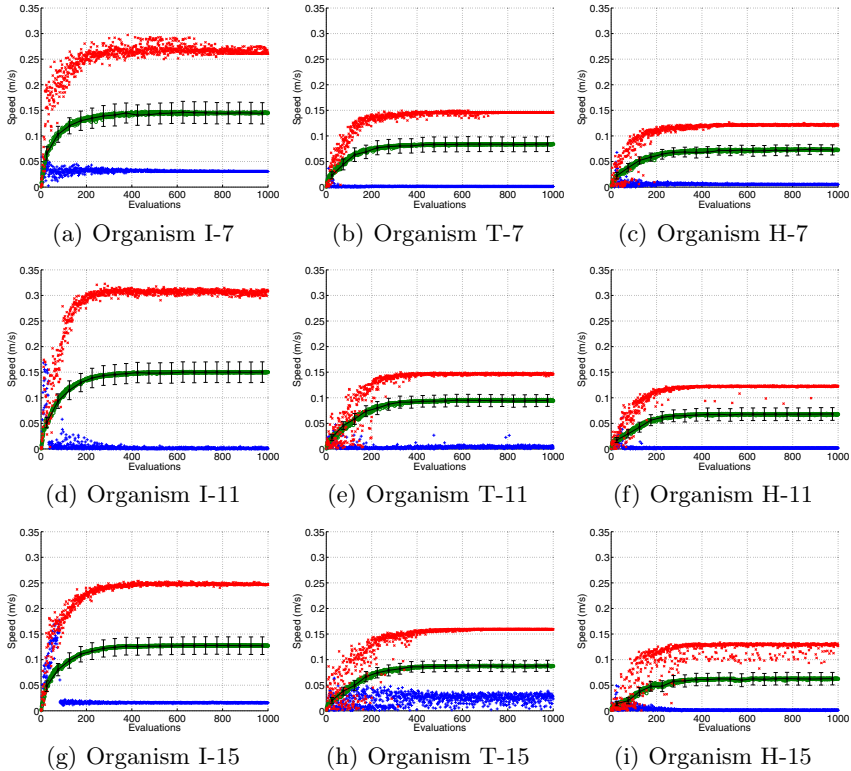
The algorithm operating parameters used for the variance and its decay factor are the same as in [16] whereas the others were chosen by hand, without further tuning. The experiment was repeated for 30 times for each organism, with different random seeds. An overview of all the parameters used in each experiment are described in Table 1.

## 4 Experimental Results

The performance of the algorithm is shown in Fig. 3, the graphs show the mean fitness of the controllers over 30 runs. We assess the algorithms' behaviour by the time it needs to converge and by the quality of the learned controllers. In Fig. 3 we can see that the algorithm converges after roughly 400 evaluations, regardless of the organisms shape or size. We also note that after convergence, the performance of the gait is very stable.

To reach this performance, a complete run took about 27 hours of simulated time, or just over 1 full day. In light of current hardware, where an operating time of over 4 hours is rare, this is still too long to be feasible in real hardware. However, as the algorithm converges in approximately one third of this time we can imagine reducing this time immensely. For instance, as mentioned in Section 3 the choice of evaluation length was conservative, it should therefore be safe to assume it is possible to reduce the evaluation time without affecting the results significantly.

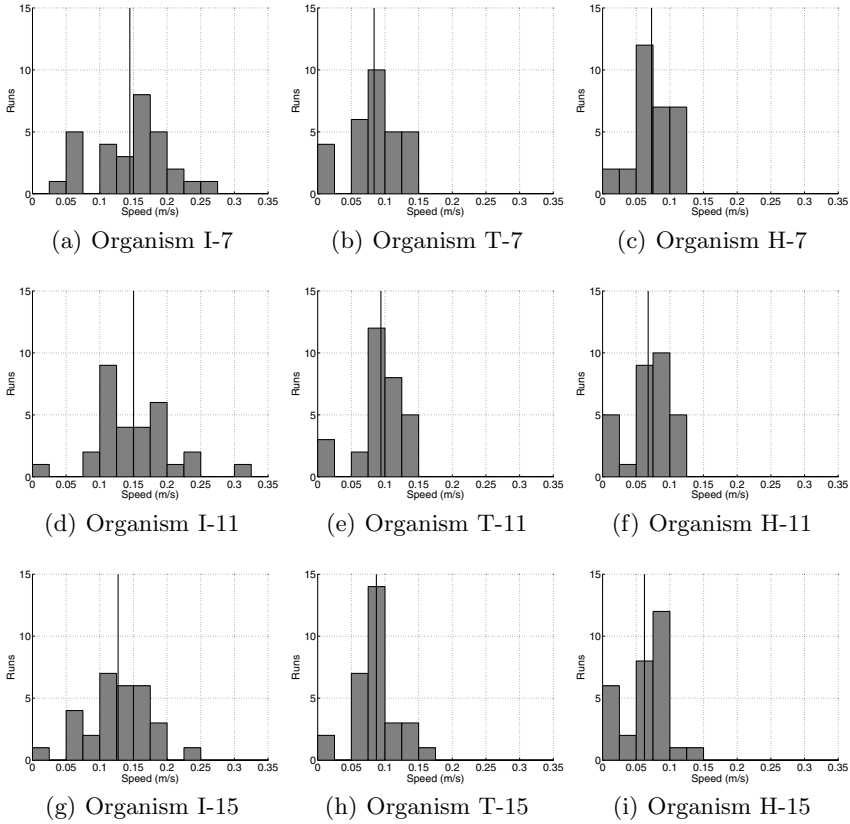
The blue curves shown in Fig. 3 display that some runs are very bad. In such runs the average speed of the organism increases until it suddenly drops



**Fig. 3.** Online dynamics of the learning process. The  $x$  axis represents time measured by the number of evaluations whereas the  $y$  axis represents evaluation performance measured by the average speed attained (m/s). The top (red) and bottom (blue) curves exhibit the best and worst single run out of the 30, respectively. The middle curves (green) show the average speed over 30 runs with the black bars representing the confidence interval.

below the 0.05 m/s mark. The reason for this huge drop is that a particular gait during the learning process made the organism lose its balance and flip on its side (I-shape), head (T-shape) or altogether (H-shape). From that point on the learning process was unable to find a good set of parameters for the now completely changed situation.

This behaviour can have two causes. First, the on-line approach used implies that between each evaluation the organism's stance or position is not reset to a default stable one. Consequently, each controller's performance is affected by the state the organism was left in by the previous one, meaning that a good move in one stance may lead to disastrous behaviour in another stance. Second, some organism morphologies may be more prone than others to lose their balance due to detrimental gaits. In [16] such detrimental gaits are filtered out based on knowledge of the organism shape and size, however, as our approach is meant



**Fig. 4.** Reliability of the learning process. The histograms show the number of runs (out of 30) terminating with a speed in a given range.

for on-line adaptation for arbitrary shapes and sizes, it is not possible to filter such gaits as we do not have knowledge of the size and shape beforehand.

All of these reasons, combined with the decreasing exploration, may have led to such bad runs. In the Triangle of Life, however, bad runs should not be a big problem, shapes which are prone to bad runs will not be able to reproduce and hence these shapes disappear over time. On the other hand, shapes that are well suited for balanced locomotion will be able to prevail.

The histograms in Fig. 4 show the distribution of runs based on the performance of the last controller, the black vertical bar represents the mean. The distribution of runs allows us to determine the reliability of the learning process by the number of bad runs for each organism. These histograms show quite clearly for each organism the number of runs of which the outcome is a controller with a very low average speed (below 0.05 m/s). Over the different experiments there is a minimum of 1 bad run and a maximum of 7 bad runs, which corresponds to a percentage between 3.3% and 23.3%.

**Table 2.** T-Test Results - Size. The table show the results of the independent samples T-Tests comparing the performance of the last controller on organisms with same complexity but different size. **NS** means the T-test showed the difference is not significant, the corresponding p values are included.

	I	T	H
7 - 11	<b>NS</b> , p = 0.710	<b>NS</b> , p = 0.300	<b>NS</b> , p = 0.531
7 - 15	<b>NS</b> , p = 0.210	<b>NS</b> , p = 0.718	<b>NS</b> , p = 0.188
11 - 15	<b>NS</b> , p = 0.094	<b>NS</b> , p = 0.423	<b>NS</b> , p = 0.537

**Table 3.** T-Test Results - Complexity. The tables show the results of the independent samples T-Tests comparing the performance of the last controller on organisms with same size but different complexity. **NS** means the T-test showed the difference is not significant, **S** means the difference *is* significant, the corresponding p values are included.

	7	11	15
I - T	<b>S</b> , p = $1.602 \times 10^{-05}$	<b>S</b> , p = $1.821 \times 10^{-05}$	<b>S</b> , p = $2.873 \times 10^{-04}$
I - H	<b>S</b> , p = $7.959 \times 10^{-08}$	<b>S</b> , p = $5.472 \times 10^{-09}$	<b>S</b> , p = $1.119 \times 10^{-07}$
H - T	<b>NS</b> , p = 0.217	<b>S</b> , p = 0.005	<b>S</b> , p = 0.004

The number of bad runs seems to be quite constant among organisms of the same complexity with different sizes, the I shape for instance has very few bad runs regardless of whether its size is 7 or 15 modules. On the other hand across complexity there does seem to be an influence on the number of bad runs: the T and H shapes have relatively more bad runs than the I shape. More experiments need to be carried out in order to assess if this relationship between the number of bad runs and organism complexity holds more generally. Going back to the performance of the organisms, Fig. 3 shows that organisms with same complexity but different size are very similar in performance. Organisms with the same size but different complexity, however, show a very large difference in performance. To see if these differences are statistically significant T-Tests with a significance level of ( $p < 0.01$ ) were conducted using the performance of the last controller of each run. The results are shown in Tables 2 and 3. There we can see that indeed the difference in performance between organisms of the same complexity, but with different sizes is not significant. The difference in performance between organisms of the same size, but different complexity is significant in most cases. This strengthens our belief that the design space for robot organisms consisting of modular robots is an interesting one.

## 5 Conclusions

In this paper we addressed a principal problem of *in vivo* evolution of modular multi-cellular robots. The Control Your Own Body problem arises because



newborn robot organisms are likely to have bodies and controllers that do not fit well. Therefore, every ‘baby robot’ needs to learn to control its own body. Furthermore, it needs to learn this quickly and on-the-fly by an online learning method, because Life goes on in the Triangle of Life, without a grace period. In this study we reduced this to a gait learning problem and investigated a solution by applying reinforcement learning. We conducted simulation experiments using robot morphologies of different size and complexity.

The main finding of our research is that the RL PoWER algorithm can successfully perform this learning task. Its success seems to depend more on the shape than on the size of the organisms, while its speed proved rather independent from either of these factors. The differences between morphologies can be quite substantial: The I shape had very few bad runs where the organism fails to move at all, whereas the failure rate for the H shape can be up to 20%. These failures are due to very bad gaits that cause the organism to lose its balance and flip on its side or back.

Regarding the speed of the learning process we found that the learning algorithm converges to the best gait after around 400 evaluations for all shapes and sizes we tested here. In terms of time, our experiments completed in roughly 27 simulated hours for 1,000 evaluations, where each evaluation ran for 96 seconds. However, we expect that the evaluations could be made shorter without losing much performance as the evaluation time of 96 seconds is rather long compared to other gait learning algorithms. Using shorter evaluations and stopping the learning process after 400 evaluations we could reduce the learning period to approximately 2 hours simulated time.

Further work will be carried out along three lines. First of all, we want to improve the RL PoWER algorithm by tuning its parameters. Furthermore, we are interested in comparing this method to other methods, for instance HyperNEAT. Last but not least, we want to validate our results by replicating these experiments using real hardware.

## References

1. Bongard, J., Zykov, V., Lipson, H.: Resilient machines through continuous self-modeling. *Science* 314(5802), 1118–1121 (2006)
2. Christensen, D.J., Larsen, J.C., Støy, K.: Fault-tolerant gait learning and morphology optimization of a polymorphic walking robot. *Evolving Systems* (to appear, 2013)
3. Christensen, D.J., Schultz, U.P., Støy, K.: A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots. *Robotics and Autonomous Systems* 61(9), 1021–1035 (2013)
4. Clune, J., Beckmann, B.E., Ofria, C., Pennock, R.T.: Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In: *IEEE Congress on Evolutionary Computation, CEC 2009*, pp. 2764–2771. IEEE Press (2009)
5. Eiben, A.E., Bredeche, N., Hoogendoorn, M., Stradner, J., Timmis, J., Tyrrell, A., Winfield, A.: The triangle of life: Evolving robots in real-time and real-space. In: Lió, P., Miglino, O., Nicosia, G., Nolfi, S., Pavone, M. (eds.) *Advances in Artificial Life, ECAL 2013*, pp. 1056–1063. MIT Press (2013)

6. Eiben, A.E., Kernbach, S., Haasdijk, E.: Embodied artificial evolution. *Evolutionary Intelligence* 5(4), 261–272 (2012)
7. Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Booth, M., Rossi, F.: *Gnu Scientific Library: Reference Manual*. Network Theory Ltd. (2009)
8. Haasdijk, E., Eiben, A.E., Karafotias, G.: On-line evolution of robot controllers by an encapsulated evolution strategy. In: *IEEE Congress on Evolutionary Computation, CEC 2010*, pp. 1–7. IEEE Press (2010)
9. Haasdijk, E., Rusu, A.A., Eiben, A.E.: HyperNEAT for locomotion control in modular robots. In: Tempesti, G., Tyrrell, A.M., Miller, J.F. (eds.) *ICES 2010*. LNCS, vol. 6274, pp. 169–180. Springer, Heidelberg (2010)
10. Hamann, H., Stradner, J., Schmickl, T., Crailsheim, K.: A hormone-based controller for evolutionary multi-modular robotics: From single modules to gait learning. In: *IEEE Congress on Evolutionary Computation, CEC 2010*, pp. 1–8. IEEE Press (2010)
11. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks* 21(4), 642–653 (2008)
12. 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)
13. Kamimura, A., Kurokawa, H., Yoshida, E., Tomita, K., Kokaji, S., Murata, S.: Distributed adaptive locomotion by a modular robotic system, M-TRAN II. In: *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2004*, vol. 3, pp. 2370–2377. IEEE Press (2004)
14. Kober, J., Peters, J.: Learning motor primitives for robotics. In: *IEEE International Conference on Robotics and Automation, ICRA 2009*, pp. 2112–2118. IEEE Press (2009)
15. Möckel, R., Jaquier, C., Drapel, K., Dittrich, E., Upegui, A., Ijspeert, A.: YaMoR and Bluemove – an autonomous modular robot with Bluetooth interface for exploring adaptive locomotion. In: Tokhi, M.O., Virk, G.S., Hossain, M.A. (eds.) *Proceedings of the 8th International Conference on Climbing and Walking Robots, CLAWAR 2005*, pp. 685–692. Springer (2006)
16. Shen, H., Yosinski, J., Kormushev, P., Caldwell, D.G., Lipson, H.: Learning fast quadruped robot gaits with the RL PoWER spline parameterization. *Cybernetics and Information Technologies* 12(3), 66–75 (2012)
17. Shen, W.-M., Salemi, B., Will, P.: Hormones for self-reconfigurable robots. In: Pagello, E., Groen, F., Arai, T., Dillman, R., Stentz, A. (eds.) *Proceedings of the 6th International Conference on Intelligent Autonomous Systems (IAS-6)*, pp. 918–925. IOS Press (2000)
18. Spröwitz, A., Moeckel, R., Maye, J., Ijspeert, A.J.: Learning to move in modular robots using central pattern generators and online optimization. *The International Journal of Robotics Research* 27(3-4), 423–443 (2008)
19. Yim, M.: A reconfigurable modular robot with many modes of locomotion. In: *Proceedings of International Conference on Advanced Mechatronics*, pp. 283–288. Japan Society of Mechanical Engineers, Tokio (1993)
20. Yosinski, J., Clune, J., Hidalgo, D., Nguyen, S., Zagal, J., Lipson, H.: Evolving robot gaits in hardware: the HyperNEAT generative encoding vs. parameter optimization. In: Lenaerts, T., Giacobini, M., Bersini, H., Bourguine, P., Dorigo, M., Doursat, R. (eds.) *Advances in Artificial Life, ECAL 2011*, pp. 890–897. MIT Press (2011)

# Approach for Recognizing Allophonic Sounds of the Classical Arabic Based on Quran Recitations

Yahya O. Mohamed Elhadj<sup>1</sup>, Mansour Alghamdi<sup>2</sup>, and Mohamed Alkanhal<sup>2</sup>

<sup>1</sup> Center for Islamic and Arabic Computing  
Al Imam Mohammad ibn Saud Islamic University Riaydh, Saudi Arabia  
yelhadj@ariscom.org

<sup>2</sup> Computer Research Institute, KACST, Riaydh, Saudi Arabia  
{mghamdi, alkanhal}@kacst.edu.sa

**Abstract.** In this paper we present new results related to our research work for building an accurate performance speaker independent recognizer for Classical Arabic, particularly Quran recitation; it is a part of the "Computerized Teaching of the Holy Quran" project funded by King Abdulaziz City for Sciences and Technology ([www.kacst.edu.sa](http://www.kacst.edu.sa)). In an early stage of this project, our efforts have been directed to propose a new labeling scheme covering all the Arabic sounds and their phonological variations. Since annotated speech corpus of the Classical Arabic sounds was not available yet, we next focused on building a well-designed sound database from Quranic recitations. Ten reciters have been recorded in an appropriate environment memorizing a part of the Quran resulting in almost nine hours of speech. We then concentrated on its manual segmentation and labeling on three levels: word, phoneme, and allophone in order to be accurate as much as possible due to the specificity of the Classical Arabic pronunciation and particularly the holy Quran recitation. This paper describes the development of a recognizer for the allophonic sounds of Classical Arabic (language of the holy Quran) based on Quranic recitations. This recognizer is built using the Cambridge HTK tools. Each allophonic sound is modeled by an acoustic Hidden Markov Model (HMM) with 3-emitting states. A continuous probability distribution using 16 Gaussian mixture distributions is used for each emitting state. Results show that recognition rates reached a good level of accuracy (88% average) without using any specific language model, which is very promising and encouraging.

**Keywords:** speech recognition, HMM, Arabic, speech corpus, Quran, Arabic pronunciation, Quran recitation.

## 1 Introduction

Many research efforts have been deployed to tackle the problem of Automatic Speech Recognition (ASR) using different techniques. An enormous progress has been achieved over the past decades by the use of statistical approach, namely Hidden Markov Models (HMM) [1-4].

The success of using HMM for ASR has induced the development of ASR engines to help manipulating HMMs efficiently and to accelerate the development process. Hidden Markov model Tool Kit (HTK)<sup>1</sup> and Sphinx<sup>2</sup> are the famous ASR engines used by the scientific committee in this domain. HTK is a portable toolkit developed at Cambridge University for building and manipulating hidden Markov models to develop speech recognition systems [5, 6]; it consists of a library modules and a set of more than 20 tools. Sphinx is a speech recognition engine built for HMM-based Large Vocabulary Continuous Speech Recognizers (LVCSR); it is developed at Carnegie Mellon University [7, 8].

The success achieved by HMM-based ASR and the availability of recognition engines have encouraged the development of many kinds of applications that use speech as input or output modules for different languages. These applications cover wide range of areas, such as remote control applications, handicap applications, Authentication, entrainment, indexing, speech translation, language learning, etc.

One important and distinguished application of general Arabic Speech recognition is the automation of learning the holy Quran. Although recited Quran is not used in communication, it is extremely important in teaching the pronunciation of Classical Arabic sounds (language of the holy Quran). Moreover, teaching how to pronounce Quranic sounds is indispensable for correct reading of the holy Quran, either for memorizing it or for reciting it in Islamic worshipping such as prayers.

In this paper, we present the development of an HMM-based recognizer for allophonic sounds of the Classical Arabic based on Quarnic recitations. This will consider a good background and open the door for a mint of applications for teaching different aspects in the Classical Arabic, either for the old Arabic poetry or for the available religious heritage.

## 2 An In-house Developed Sound Database: Overview

The first stage of the "Computerized Teaching of the Holy Quran" project [9] were devoted to the development of a well designed database for Quranic sounds. We started proposing a new labeling scheme covering both the Classical Arabic (language of the Quran and old poetry) as well as the Modern Standard Arabic (used nowadays in official communications and news media) [10]. Table 1 gives an extract of the Classical Arabic allophones of the single phonemes (not geminate). The first number of the code is always 1 to represent the single allophones. However, it can be 2 to represent the geminate consonants and vowels or 4, 7 or 8 to represent the longer vowel duration (called in Arabic *mudoud*). The second number is always 1 or higher to cover the allophones, not only in the Classical Arabic but also Modern Standard Arabic.

A phoneme is an abstract concept that can be manifest in different allophones. For example, the phoneme /a:/ is a long low vowel which its length varies according to its position in a word and the adjacent phonemes. /a:/ is almost double the duration of

---

<sup>1</sup> <http://htk.eng.cam.ac.uk>

<sup>2</sup> <http://www.speech.cs.cmu.edu/sphinx/Sphinx.html>

that of its short counterpart /a/ [11] in words such as: /sa:d/ "dominated" and /sadd/ "dam", respectively. However, the same vowel, /a:/ is produced even longer in a word such as /ma:ʔ/ "water" where some reciter might have it four or six times longer than its short counterpart /a/. These variations in pronouncing the same phoneme are called allophones. Some of Arabic sounds may have the same place of articulation but different manner of articulation. Sounds that are produced at the alveolar ridge can be plain stops similar to the English /t, d/, emphatic stops where the back of the tongue is retracted upwards and backwards, velarized where the tongue is drawn upwards, or nasalized where the air escapes through the nose.

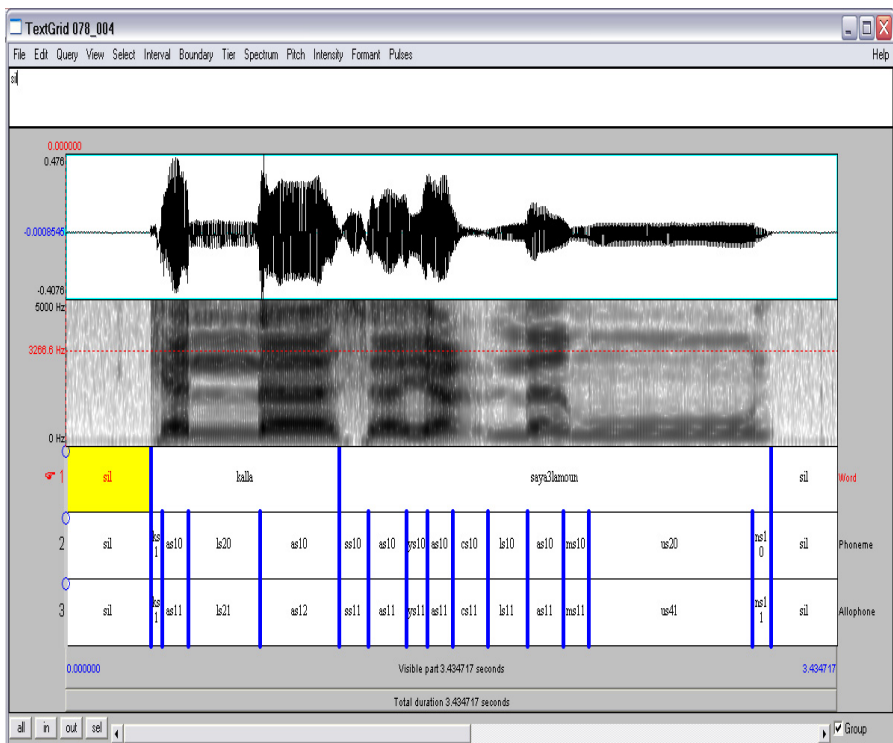
In the second stage, we collected Quranic recitations from ten chosen reciters who read verses from Quran. The recording was done in a soundproof chamber at King Abdulaziz City for Sciences and Technology (KACST). This was resulted in almost nine hours of speech with an average of around 50 minutes per reciter. A total number of 5935 sound files were obtained with an average of about 594 files per reciter. Sound files are coded on the form SSS\_XXX\_YYY\_ZZ, where SSS represents initial letters of the reciter's name, XXX represents *sourah* (Quranic chapter), YYY represents *ayah* (verse of a Quranic chapter), and ZZ represents pause numbers made

**Table 1.** Extract of some Arabic phonemes, their symbols in International Phonetic Alphabet (IPA), the used labels for allophonic variations and their phonetic description [10]

Arabic Character	IPA	Labels	Phonetic Description
ا	a	as11	Plain
		as12	emphatic
		as13	velarized
		as16	centralized
أ	u	us11	Plain
		us12	emphatic
		us13	velarized
إ	i	is11	Plain
		is12	emphatic
		is13	velarized
هـ	-	hz11	Plain
ب	b	bs11	Plain
		bs15	released with a schwa
ت	t	ts11	Plain
		ts14	nasalized
		ts15	aspirated
ث	>	vs11	Plain
		vs14	nasalized
ج	ʤ	jb11	Plain
		jb14	nasalized
		jb15	released with a schwa

by the reciters within an *ayah*. Text files of the spoken counterparts were created and named in the same manner. Preliminary versions in Arabic of the corpus were described in [12, 13]; a full version in English is currently under publication process by the Language Resources and Evaluation Journal<sup>3</sup>.

Having building the sound database, we focused on its manual segmentation and labeling on three levels: word, phoneme, and allophone using the Praat tools<sup>4</sup> (see figure 1). Labels of words are composed of *sourah* number, *ayah* number, and word number joined by underscores. The transcription files are extracted from PRAAT and kept in their original format as "txtgrids" files to allow an easy manipulation at further stages. Each sound file has now a corresponding file that represents its transcription (.txtgrid) in addition to the text file containing the corresponding text.



**Fig. 1.** Example of the segmentation and labeling on three levels: word, phoneme, allophone

<sup>3</sup> <http://www.springer.com/education+%26+language/linguistics/journal/10579>

<sup>4</sup> <http://www.fon.hum.uva.nl/praat/>

### 3 Development of the Recognizer

#### 3.1 Dictionary of Pronunciation (lexicon) and List of Allophonic Sounds

We have used transcription files to extract all reciters pronunciations of words as occurred in the sound files (see table 2 for an extract). It is worth to mention that, at the best of our knowledge, there is no available dictionary which can be used to obtain Arabic words pronunciation (especially the Classical one and thus Quranic words) as it is for other languages such as English for example [14].

The number of unique words in the textual part of our corpus (last part of the Quran) was 1377 for which we added a silence (SIL) as special word to represent the start and the end of *ayahs* at the word level; with repetition, this number reaches 2348. The total number of words for all reciters is 35353, from which 11873 represent the silence, and the remaining 23480 represent the real words (2348 words for each one of the 10 reciters). The statistics show that the majority of reciters have almost the same number of pronunciations, which reaches 2260, giving an average of about 1.7 pronunciations per word. From that, we can see in Table 2 that almost all words have between one and two pronunciations; a few words have three pronunciations.

Table 2. Extract from the pronunciation dictionary

cs11 as11 ys11 ns11 us11 ns21 عَيْنٌ	hz11 as11 hz11 is11 vb11 as21 نَيْدًا
cs11 as11 ys11 ns11 is11 ns11 عَيْنٌ	hz11 as11 hz11 is11 ns21 as21 أَنْثًا
cs11 as11 ys11 ns11 as11 عَيْنٌ	hz11 as11 hz11 as11 ns21 ts14 us11 ms11 أَنْثَمٌ
cs11 as11 ys11 ns11 as11 ys24 عَيْنًا	hz11 as11 bs11 as21 bs11 is41 ls11 أَبَايِلٌ
cs11 as11 ys11 ns11 as11 ys21	bs15 ts11 is11 gs11 as42 hz11 as11 ابْتِغَاءٌ
gs11 as12 bs11 as11 rs11 as12 عَجْرَةٌ	bs15 ts11 is11 gs11 as61 hz11 as11
hs11	bs15 ts11 is11 gs11 as62 hz11 as11
gs11 us11 vs11 as61 hz11 as11 عَشَاءٌ	bs15 ts11 as11 ls11 as21 hs11 us11 ابْتِلَاءٌ
ns11	hz11 as11 bs11 as11 ds11 as21 ابْتِدَاءٌ
gs11 us11 vs11 as41 hz11 as11	hz11 is11 bs15 rs12 as22 hs11 is21 ms11 ابْتِرَاهِيمَ
ns11	as11
gs11 as12 rs12 qs11 as22 عَرَفًا	hz11 as11 bs15 sb11 as12 rs12 us12 ابْتِصَارُهَا
gs11 as11 rs11 qs11 as21	hs11 as21
gs11 as12 rs22 as12 ks11 as11 عَرَكَ	hz11 as11 bs15 ws11 as21 bs11 as21 ابْتَوَايَا
gs11 as12 ys11 rs11 us11 عَجْرٌ	hz11 as11 bs11 is21 أَبِي
gs11 as12 ys11 rs12 us12	hz11 as11 ts11 as21 ks11 as11 أَتَاكَ
	hz11 as11 ts15 rs12 as22 bs11 as41 أَتْرَابًا
	HZ11 as11 ts15 rs12 as22 bs11 as21

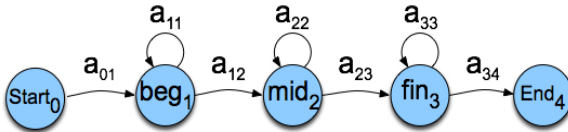
From the transcription files, we also extracted the list of unique allophonic sounds (see table 3 for a part of them). They count 109 plus the silence unit (sil) which is counted as normal allophone to represent short pauses during the recitations as well as the start and the end of *ayahs*.

**Table 3.** List of the Allophones from sound files

as11	bs21	ds21	is11	jb15	ls12	qs15	sil	ts15	vb14	ys11
as12	bs25	fs11	is12	jb21	ls21	qs21	ss11	ts21	vb21	ys14
as21	cs11	fs14	is21	js11	ls22	qs25	ss14	us11	vs11	ys21
as22	cs21	fs21	is22	js14	ms11	rs11	ss21	us12	vs14	ys24
as41	db11	gs11	is41	js21	ms21	rs12	tb11	us21	vs21	zb11
as42	db14	hb11	is42	ks11	ms24	rs21	tb14	us22	ws11	zb14
as61	db21	hs11	is61	ks14	ns11	rs22	tb15	us41	ws21	zb21
as62	ds11	hs21	is62	ks15	ns21	sb11	tb21	us42	ws24	zs11
bs11	ds14	hz10	jb11	ks21	qs11	sb14	ts11	us61	xs11	zs14
bs15	ds15	hz11	jb14	ls11	qs14	sb21	ts14	vb11	xs21	zs21

### 3.2 Acoustic Modeling and Features Extraction

We used Hidden Markov Models [15] to represent allophonic sounds of the Classical Arabic. Recent studies indicate that an HMM with three emitting states (corresponding to the transition-in, steady state, and transition-out regions of the phone) is generally enough to capture the acoustic properties of a particular sound [4]. Figure 2 shows a typical 3-emitting state HMM architecture (taken from [4]), which we adopted for our allophonic sounds.

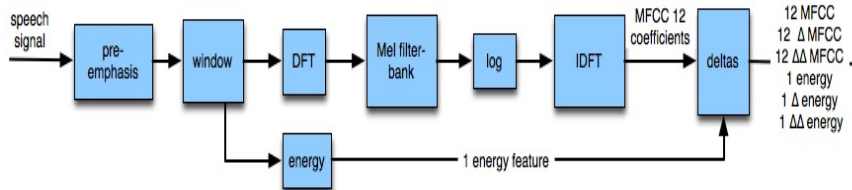
**Fig. 2.** Adopted HMM architecture for allophonic sounds

A Gaussian Mixture Models (GMM) will be associated to each state of the indicated HMM acoustic model to identify the characteristics of the sound portion at this state.

Once the architecture of acoustic model is defined, the structure of feature vectors has to be specified. This means that an input wave-form is transformed to a sequence of feature vectors, each of them representing the signal in a small time window; usually Hamming window with size 10 to 30 ms. Among the different possible parameterizations of signal, MFCC (Mel Frequency Cepstral Coefficients) is the most known and popularly used for speech processing. Current ASR systems limit the number of MFCC coefficients considered to the first twelve ones representing the static features of the signal portion, for which the first and second derivatives (velocity and



acceleration) are added to capture the dynamic features. Energy of the signal portion and its derivatives are also considered, and thus a feature vector of 39 coefficients is obtained. Figure 3 shows the standard MFCC extraction steps taken from [4].



**Fig. 3.** Feature extraction process

Our sound corpus was recorded with high sampling rate (44100 Hz and 16 bits) to be useful for a variety of applications other than speech recognition, which generally necessitate no more than 20000 Hz. We have sampled-down a version of it, using appropriate software, to be used in the development of the current recognizer. We used the library HCopy of the HTK tool-kit [6] to extract feature vectors from our speech signals at a rate of 8 ms within a Hamming window of size 16 ms. Thus, a vector of 39 coefficients (as previously described) is generated every 8 ms.

### 3.3 Training

The training phase aims to inject sound features in the parameters of underlying acoustic models with the goal of becoming able to recognize them at a later stage. Training can be done in isolated-unit or embedded-unit mode depending on the type of recognition task. Isolated-unit training is usually employed for separated unit-models or at least with clear speech-boundaries as in a word-based modeling task for example; while the embedded-unit training is typically used for sub-word (e.g. phone) modeling in a continuous speech recognition task. Isolated-unit training might be important for embedded-unit training if we have enough boundaries-marked data. However, having a boundaries-marked of huge amount of data (usually required for VLCSR systems) is not always obvious and might not be practically feasible.

In the HTK, isolated-unit and embedded-unit training are respectively performed using HInit followed by HRest, and HCompV followed by HERest. HInit and HCompV are used to initialize models respectively in the isolated-unit mode (where models are separated or with known boundaries) and embedded-unit training mode (where boundaries might not be clear). HRest and HERest are used to perform the proper training using the Baum-Welch procedure. In the case of sub-word modeling with marked-boundaries, HInit and HRest can be used for initialization in the embedded-unit training and thus might be regarded as a bootstrap operation. In our case, we will start using isolated-unit training as our sound database is manually segmented and labeled at different levels, among them the allophonic level. Embedded-unit training will then be conducted to ensure a uniform training. Moreover, we will test different possible combinations of the above HTK training tools to select the best appropriate one.

### 3.4 Language Modeling

In this work, we used a flat language model on the level of allophones to allow all possible combination of pronounced sounds. A specific language model may impose or guide the recognizer to predict some allophones, and thus to consider them correctly pronounced. Table 4 gives the simple language model we used.

**Table 4.** Language Model for Allophonic Sounds

<pre> \$All = sil   hz11   as11   cs11   us21   vb11   us11   bs11   is11   ls21   as21   hs11   ms11   ns11   js21   ys11   tb11   as22   rs22   jb11   is21   ss11   as12   hb11   ls11   ds11   rs12   bs21   is41   ks11   ws11   ds21   ys21   ts11   sb21   is12   qs11   is42   sb11   gs11   rs11   db11   us22   db21   as62   ms21   as41   us41   ns21   zb11   is22   fs11   xs11   vs11   jb15   ts15   qs15   zs11   js11   bs15   js14   ws24   hs21   zs14   as61   vs14   jb21   ... ; ( &lt; \$All &gt; ) </pre>
--

The HTK tool called HParse is used to transform this language model to an internal format called Standard Lattice Format (SLF) to be used in the recognition phase by the tool HVite, which is based on the library HRec.

### 3.5 Experimentations

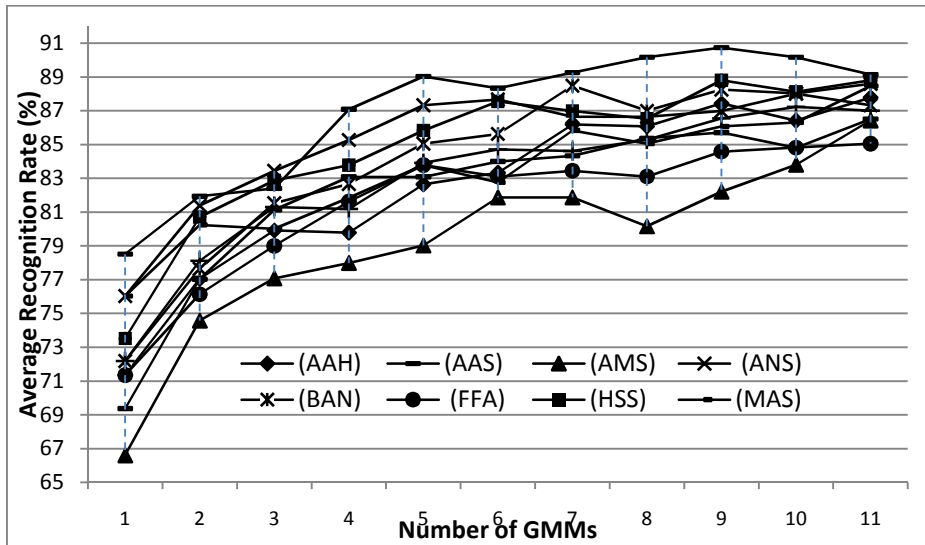
The sound database is divided into ten groups of training and testing sets; all of them are used in our main experimentations one at a time and then a global average is computed. For each group, we consider a particular reciter to construct the testing set by extracting the first ayah of each *sourah* from it; the remaining ayahs of this reciter as well as all ayahs of the other reciters are used for training. Since our sound database contains 38 *sourahs* and 572 *ayahs*, so in each group the training and testing sets are respectively composed of 534 and 38 *ayahs*. This means that in each group, about 93% of the corpus is used for training and 7% is used for testing.

Different kinds of experimentations were conducted. The first type was to select the best combination of HTK training tools to ensure a good training of the models. We start using HInit + HRest and then HCompV + HERest; we noticed that the best results were obtained from the first combination. This is somehow expected as our manual segmentation was done with an acceptable level of accuracy. So, guided by this inspiration that HInit gives a good initialization and thus could be considered as starting point, we conducted two other experimentations, one using HInit + HERest, and another one using HInit + HRest + HERest. And we finally got the best results with the last combination. So, all the following experimentations will be conducted using this combination (HInit + HRest + HERest) for training. The second type of experimentations was focused on the combination of GMMs to determine the optimal number appropriate for our sound database. We conducted a lot of experimentations varying the number of GMMs from 1 to more than 16 on the previously mentioned

groups of training & testing sets (more than 160 experimentations). We report in the table 5, the results from 1 to 16 GMMs as the higher numbers did not show significant recognition rates. These results are plotted in the figure 4..

**Table 5.** (%) of Recognition Rates for 1 to 16 GMMs

GMM	All the ten Reciters (letters are the initials of their names)									
	AAH	AAS	AMS	ANS	BAN	FFA	HSS	MAS	MAZ	SKG
1	71.35	78.51	66.59	76.03	72.18	71.35	73.52	76.03	69.37	72.18
2	77.05	81.94	74.57	81.39	77.65	76.14	80.71	80.25	77.03	78.11
3	79.91	82.4	77.08	83.45	81.53	79	82.88	80.02	81.14	81.3
4	79.79	87.09	77.99	85.27	82.67	81.62	83.79	81.85	83.09	81.19
5	82.65	89.03	79.02	87.33	85.06	83.79	85.84	83.79	83.09	83.92
6	83.33	88.34	81.87	87.67	85.63	83.11	87.56	82.76	84	84.72
7	86.19	89.26	81.87	86.64	88.48	83.45	86.99	85.84	84.34	84.61
8	86.07	90.17	80.16	86.64	87	83.11	86.53	85.05	85.37	85.29
9	87.44	90.74	82.21	86.99	88.26	84.59	88.81	86.07	85.71	86.55
10	86.42	90.17	83.81	88.01	88.03	84.82	88.13	86.3	84.8	87.23
16	87.79	89.14	86.43	87.33	88.6	85.05	88.81	88.47	86.51	87



**Fig. 4.** % of Recognition Rates for different GMMs

## 4 Results Discussion

The first thing to be noted is that the recognition rates are not bad as the poorest one is about 67% for the reciter AMS with 1 GMM and the average is 73%, which is an acceptable rate. The second remark is the neat improvement of recognition rates obtained when increasing the number of GMMs, as the lowest recognition rate reached 85% and the average is 88%, which represent a global enhancement of about 21%. This improvement varies from reciter to another depending on both training & testing groups and the number of GMMs. Thus, it was 12% for the reciter ANS and 20% for the reciter MAZ. The reciter AMS, which held previously the lowest score, is now gained the best improvement 20%; this could be explained by the fact that sound files of this reciter may be were not so clean, but with the increment of GMMs, sounds become distinguishable and thus being neutralized. The third remark is that almost the best results are obtained by the first ten GMMs and that beyond this number no significant improvements was noted. This seems to be very logical as our data is clean (at a great level) and thus ten GMMs were good enough to represent the variability of ten reciters. An in-depth analysis conducted on the confusion matrix show that allophonic variations of some sounds were confused with others, especially the lengthening degrees (*mudoud*).

## 5 Conclusions and Future Directions

In this paper, we presented the development of a recognizer for allophonic sounds related to the Classical Arabic pronunciation. These sounds are extracted from Quranic recitations as it is the general medium actually representing Classical Arabic. This recognizer is based on a well designed sound database composed of recitations of a part of the holy Quran recorded from ten chosen reciters. Speech signals were manually segmented and labeled on three levels, words, phonemes, allophones, with a good precision and accuracy.

HTK was used to build the recognizer and several experimentations were conducted to tune its performance. The results highlight the importance of the approach developed and show good recognition rates. However, we still think that the obtained recognition rates could be improved by working on different directions, such as for examples: 1) modeling the context between allophones by introducing triphones; 2) looking for a more specialized HMM architectures for allophonic sounds with the aim to distinguish between them; 3) limit the recognizer to the level of phonemes (their number is about 60) and in a post processing phase use specific characteristics of sounds (e.g. duration) to distinguish allophonic variations. This last trend is currently being developed and might be very promising as some allophonic variations of Quranic sounds tend to be very close from each other and seems difficult to be distinguished by a typical approach with HTK.

**Acknowledgements.** This paper was supported by King Abdulaziz City for Science and Technology under the grant number AT-25-113, Riyadh, Saudi Arabia. We thank all the members of our project team.

Some of the experimentations presented here had been conducted during a research stay within the SAMOVA team, IRIT, Toulouse; we thank Mrs R. André-Obrecht, Mr. J. Farinas, Mr. J. Pinquier for their creative remarks and suggestions.

## References

1. Rabiner, L., Juang, B.H.: *Fundamentals of Speech Recognition*. Prentice Hall (1993)
2. Jelinek, F.: *Statistical Methods for Speech Recognition*. MIT Press, Cambridge (1998)
3. Huang, X., Acero, A., Hon, H.: *Spoken Language Processing*. Prentice Hall PTR (2001)
4. Jurafsky, D., Martin, J.H.: *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*, 2nd edn. Prentice-Hall (2008)
5. Young, S.: *The HTK hidden Markov model toolkit: design and philosophy* (Tech. Rep. CUED/FINFENG/, TR152). Cambridge University Engineering Department, UK (September 1994)
6. Young, S., et al.: *The HTK Book* (Version 3.4). Cambridge University Engineering Department (2009)
7. Huang, X., Allewa, F., Hon, H.W., Hwang, M.Y., Rosenfeld, R.: *The SPHINX-II speech recognition system: an overview*. *Computer Speech and Language* 7(2), 137–148 (1993)
8. Rabiner, L., Juang, B.H.: *Fundamentals of speech recognition*. Prentice-Hall, Englewood Cliffs (1993)
9. Elhadj, Y.O.M., Alsughayeir, I.A., Alghamdi, M., Alkanhal, M., Ohali, Y.M., Alansari, A.M.: *Computerized teaching of the Holy Quran (in Arabic)*, Final Technical Report, King Abdulaziz City for Sciences and Technology (KACST), Riyadh, Saudi Arabia (2012)
10. AlGhamdi, M., Elhadj, Y.O.M., AlKanhal, M.: *A manual system to segment and transcribe Arabic Speech*. In: *Proceedings of IEEE ICSPC 2007*, Dubai, UAE, pp. 233–236 (2007) ISBN 1-4244-1236-6
11. Alghamdi, M.: *Analysis, Synthesis and Perception of Voicing in Arabic*. Al-Toubah Bookshop, Riyadh (2004)
12. Elhadj, Y.O.M., AlGhamdi, M., AlKanhal, M., Alansari, A.M.: *Sound Corpus of a part of the noble Quran (in Arabic)*. In: *Proc. of the International Conference on the Glorious Quran and Contemporary Technologies*, King Fahd Complex for the Printing of the Holy Quran, Almadinah, Saudi Arabia, October 13-15 (2009)
13. Elhadj, Y.O.M.: *Preparation of speech database with perfect reading of the last part of the Holly Quran (in Arabic)*. In: *Proc. of the 3rd IEEE International Conference on Arabic Language Processing (CITAL 2009)*, Rabat, Morocco, May 4-5, pp. 5–8 (2009)
14. Carnegie Mellon University. *The Carnegie Mellon Pronouncing Dictionary v0.1* (1993)
15. Rabiner, L.: *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. *Proceedings of the IEEE* 77(2) (1989)

# Reliable Self-assembly by Self-triggered Activation of Enveloped DNA Tiles

Vinay Kumar Gautam<sup>1</sup>, Pauline C. Haddow<sup>1</sup>, and Martin Kuiper<sup>2</sup>

<sup>1</sup> CRAB Lab, Gemini Centre for Applied Artificial Intelligence  
Department of Computer and Information Science

{vkgautam,pauline}@idi.ntnu.no

<sup>2</sup> Semantic Systems Biology Group

Department of Biology

The Norwegian University of Science and Technology

Trondheim, Norway

martin.kuiper@ntnu.no

**Abstract.** Although the design of DNA tiles has been optimised for efficient and specific self-assembly, assembly errors occur so often that applications for molecular computation remain limited. We propose the use of an enveloped tile consisting of a DX- base tile that carries a protector tile to suppress erroneous tile assembly. The design of the enveloped tile promotes the dissociation of the protector tile from the base tile through a self-triggered activation process, which keeps the outputs of the base tile stay protected until both base tile inputs have bonded correctly to the assembly. The enveloped tile design, the self-triggered activation that removes the protector tile and preliminary modelling results are presented.

**Keywords:** Enveloped Tile, Self-assembly, DNA tile, Molecular computation, Protector Tile.

## 1 Introduction and Motivation

DNA tile self-assembly [9] — the emergence of assemblies through physicochemical interactions between basic building blocks termed *DNA tiles*, provides a potential technology for programmable computation at the molecular scale. DNA tiles are abstract representations of DNA molecules, e.g. DNA double crossover (DX) molecule [11], with certain number of protruding sticky ends of single stranded DNAs (ssDNAs). DNA tiles may be designed with combinatorial DNA sticky ends that can bind together by canonical base pairing (A-T and G-C binding), and thereby tiles may self-assemble. However, the DNA tile assembly process is challenged by a high level of assembly errors.

There are three basic types of assembly errors: growth errors [10], facet nucleation errors [3], and spontaneous nucleation errors [8]. The first two types of errors both result from erroneous tile trapping. Growth errors occur when a tile gets trapped at a vacant site of a growing assembly despite one of its sticky

ends failing to match correctly. Facet nucleation errors appear when an incorrectly attached tile at a growing surface fails to detach quickly and eventually gets fixed. Spontaneous nucleation errors are attributed to spurious assemblies growing without a seed.

Tile designs for error prevention and correction include: redundant tile sets [5,10], protected tiles [4,6] and morphological encoding [2,12]. In redundant tile sets, each tile is replaced by a  $k \times k$  tile set e.g. proofreading tiles [10] and snake proofreading tiles [5]. However, the improved error resilience requires  $k^2$  times more tiles. Such a solution is not only disadvantaged by the increase in the necessary resource but further designing larger tile sets with orthogonal sticky ends imposes practical challenges.

The protected tile approach avoids such resource increases e.g. Protected and Layered Tiles [4] and “Activatable” Tiles [6]. However, the Protected and Layered tiles are not fully protected on their outputs — only three bases in the best case and leaving 11 bases vulnerable for spurious binding. On the other hand, the inputs and outputs of the Activatable tiles are fully protected. However, the approach requires the support of biological enzymes, which poses practical limitation.

The third approach uses morphological encoding to enhance specificity of binding i.e. correct binding between two tiles requires a match of both the binding patterns and the geometry of the involved inputs and outputs [2,12]. However, such morphological encoding poses a further design challenge due to the need for more than one sticky end at each of the inputs and outputs.

In this work, the Base Tile design remains similar to that of the well established DX-tile [11]. The assembly error resilience is thus achieved through the new Protector Tile in combination with the Base Tile termed Enveloped Tile. In particular, the design of the Protector Tile controls the self-triggered activation process resulting in either the correct attachment of the Base Tile to the assembly or release of it (if erroneous) and thus ensures directional growth of the assembly.

The remainder of the article is structured as follows: background of DNA tile self-assembly and its modelling are described in section 2. Section 3 presents proposed Enveloped Tile, and further describes design of Base Tiles and Protector Tiles for Sierpinski Triangle assembly [10]. In Section 4, self-assembly steps involving association of Enveloped Tile and self-triggered dissociation of Protector Tile has been discussed. Further, in section 5, error prevention performance of Enveloped Tile assembly has been analysed by a physically realistic kinetic model and preliminary results are documented. Section 6, concludes the article.

## 2 Background

In order to assemble uniquely defined structures or to perform a computation step using DNA tile self-assembly, a certain tile set with properly designed sticky ends is required. The tile set often consists of a seed tile, boundary tiles and rule tiles [10]. Binding strengths of sticky ends of these tiles depend on their respective

ssDNAs. Two such tiles may join together if their sticky ends are complementary and of the same length. However, such binding is a reversible chemical process, and an assembled tile is stable only if its associated total binding strength is sufficient to prevent its dissociation under local physical conditions.

The kinetic Tile Assembly Model (kTAM) [10] provides a way for mathematical modelling of the proposed tile solution. The kTAM considers each tile self-assembly step as a reversible process governed by the tile concentration, local reaction temperature and the length of the tile's sticky ends. Furthermore, it assumes that a) the tile concentration is constant for each tile type and b) only one tile can attach/detach from the growing aggregate at a time. The model enables analysis of the assembly errors and growth rate for a given tile set. Further, the tile design itself may be optimised through parameter adjustment.

Reversible kinetics of tile assembly leading to association or dissociation from a vacant site of the growing aggregate can be represented by equivalent forward and reverse reaction rates, respectively. The rate of tile attachment at a binding site of an aggregate is directly proportional to the tile concentration. The concentration of each type of tile (except the seed tile) can be given by  $e^{-G_{mc}}$ , where  $G_{mc}$  is the decrease in entropy when a tile binds at a vacant site. Therefore, the forward reaction rate ( $r_f$ ) can be given by  $r_f = k_f e^{-G_{mc}}$  where  $k_f$  is the reaction rate constant. Similarly, the tile detachment process is controlled by the energy required to break any single tile-aggregate bond and denoted by  $G_{se}$ . The value of  $G_{se}$  depends on the sticky end length ( $s$ ) and the temperature ( $T$ ), where  $G_{se} \approx (4000/T - 11)s$ . The tile reverse reaction rate involving  $b$  tile bonds is given by  $r_{r,b} = k_f e^{-bG_{se}}$ .

A larger value of  $G_{mc}$  thus implies a lower tile concentration and consequently a slower forward reaction rate (or vice versa). Similarly, a larger value of  $G_{se}$  results in a slower detachment rate. The optimum growth rate with low error rates happens near thermodynamic equilibrium ( $G_{mc} \approx 2G_{se}$ ) [10], and may be given by  $r_g \approx \frac{r_f - r_{r,2}}{2}$  and  $\varepsilon \approx e^{-G_{se}}$ , respectively. Therefore, a relation between optimum growth rate and minimum error rate may be given by  $r_g \approx \beta \varepsilon^2$  where  $\beta = 10^5$  /M/sec. Thus, any effort to reduce the error rate ( $\varepsilon$ ) by tuning physical parameters ( $G_{mc}$  and  $G_{se}$ ) would result in a quadratic reduction of the growth rate. However, error reduction without significant fall off in assembly growth rate has been achieved by adding redundant tiles [10,5] and by protecting tile's inputs and outputs [4,6].

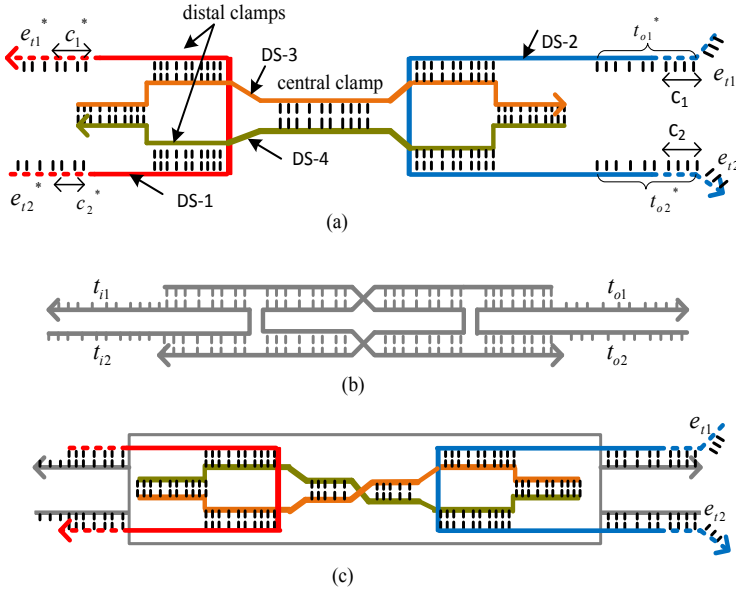
### 3 Proposed *Enveloped Tile Design*

The Base Tile (*BT*) of the Enveloped Tile (*ET*) is designed based on the DNA double crossover Anti-parallel molecule with Odd spacing (DAO molecule) [11] — an accepted building block for DNA tile self-assembly. A Protector Tile (*PT*) provides the unique tile design, assembling with the *BT* and thus protecting the *BT*'s inputs and outputs whilst leaving protruding DNA toeholds for self-triggered activation of *BT*.



### 3.1 Enveloped Tile

The centerpiece of the proposed design is the *Protector Tile* — illustrated in Figure 1(a), involving four ssDNAs (DS-1, DS-2, DS-3 and DS-4). DS-1 and DS-2 serve as protection for the *BT*'s input and output sticky ends, respectively, and are clamped together by DS-3 and DS-4. The double stranded DNA (dsDNA) sections on both sides of the central clamp act as distal clamps, linking the proofreading events at the input side to the strand displacement events at the output side.



**Fig. 1.** (a) Protector Tile (*PT*) (b) Base Tile (*BT*) and (c) Enveloped Tile (*ET*)

Each dsDNA section (depicted by vertical dashed lines) is 10 nucleotides ( $nt$ ) long and formed by two anti-parallel ssDNAs. One side of each ssDNA is marked by an arrowhead representing its 3' end, the other side of it is the 5' end. The 7  $nt$  sticky ends of DS-1 ( $e_{i1}^*$ ,  $c_1^*$  and  $e_{i2}^*$ ,  $c_2^*$ ) are complementary to the 7  $nt$  sections ( $e_{t1}$ ,  $c_1$  and  $e_{t2}$ ,  $c_2$ ), respectively. The input sticky ends of the *PT* ( $e_{i1}^*$ ,  $c_1^*$  and  $e_{i2}^*$ ,  $c_2^*$ ) of DS-1) are also complementary to the 7  $nt$  of the input sticky ends of the *BT* — see Figure 1(c). Similarly, 7 $nt$  of the output sticky ends of the *PT* ( $t_{o1}^*$  and  $t_{o2}^*$  of DS-2) are complementary to the output sticky ends of the *BT*.

The central clamp in the *PT* keeps its complementary input and output free ends apart. Based on the physics of stiffness and coiling of dsDNA [1], it is assumed that the stiffness of a 10  $nt$  dsDNA clamp will favour *PT* binding with the *BT* when the *PT* and the *BT* are allowed to bond. Such bonding occurs prior

to the *ET* being made available for the assembly process. Another important aspect of this clamp is that it will be twisted (coiled) by one turn when the *PT* is bound with the *BT* due to the 3' and 5' orientation of its input sticky ends, shown in Figure 1(c).

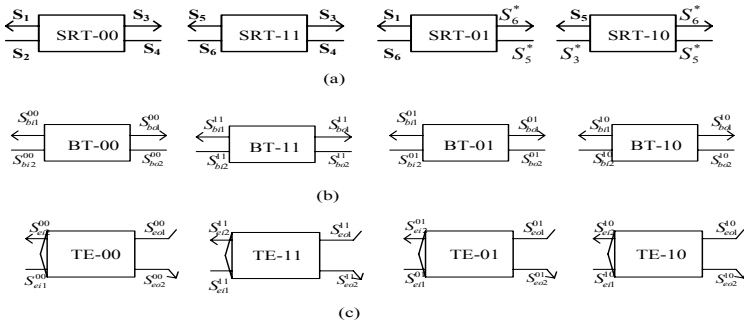
The two distal clamps play an important role in the case where only one side of the *PT* gets dissociated i.e. an erroneous tile, and by using this process no erroneous tile Assembly happens. The clamps hold the dissociated input ends close to their binding sites (emulating a locally elevated concentration of these binding sites). This in turn prevents the inputs of the *PT* from engaging with its outputs. The clamps, therefore, reinforce the reverse branch migration so that a partially displaced *PT* can return to its original structure when released from the assembly.

The Base Tile (DAO molecule) — see Figure 1(b), has 10 *nt* sticky ends ( $t_{i1}$ ,  $t_{i1}$ ,  $t_{o1}$  and  $t_{o2}$ ). Ten *nt* was selected so as to provide sufficient strength for *PT* binding whilst leaving 3 *nt* toeholds on the input side. The crossover points within the *BT* are separated by 16 *nt* (3 half-turns).

### 3.2 Enveloped Tile Set for the Sierpinski Triangle

Figure 2 illustrates a Sierpinski tile set design using Enveloped Tiles. As shown in (a), each Sierpinski Rule Tile (SRT-00, SRT-11, SRT-01 and SRT-10) has a different combination of input and output sticky ends (complementary sticky ends are marked by \*). The *BTs* are shown in (b) and the *PTs* are shown in (c). The Input and output sticky ends of the *BTs* and *TEs* are represented by  $(S_{bi1}^{jk}, S_{bi2}^{jk}), (S_{bo1}^{jk}, S_{bo2}^{jk})$ , and  $(S_{ei1}^{jk}, S_{ei2}^{jk}), (S_{eo1}^{jk}, S_{eo2}^{jk})$  respectively, where  $jk = 00, 11, 01, 10$  represents the type of Rule tile.

There are three requirements for complementary DNA sequences among these sticky end sequences — between the *BTs*' inputs and outputs; between the *BTs*' and *PTs*' inputs and outputs and between the *PTs*' inputs and outputs. Thus



**Fig. 2.** Enveloped Tile Set for Sierpinski Triangle: (a) Rule Tiles, (b)Base Tiles used for Sierpinski Enveloped Tile set (c) Corresponding Protector Tiles

**Table 1.** BTs Sticky Ends

Name	DNA sequence
$S_{bi1}^{00}$	5'-TCCTAGGACT-3'
$S_{bi2}^{00}$	3'-CTGTGTCAGG-5'
$S_{bo1}^{00}$	3'-AGGTACCTGA-5'
$S_{bo2}^{00}$	5'-GACCAAGTCC-3'
$S_{bi1}^{11}$	5'-CGTTAGGACT-3'
$S_{bi2}^{11}$	3'-GTTTGTTCAGG-5'
$S_{bo1}^{11}$	3'-AGGTACCTGA-5'
$S_{bo2}^{11}$	5'-GACCAAGTCC-3'
$S_{bi1}^{01}$	5'-CGTTAGGACT-3'
$S_{bi2}^{01}$	3'-CTGGTTCAGG-5'
$S_{bo1}^{01}$	3'-GCAATCCTGA-5'
$S_{bo2}^{01}$	5'-CAGACAGTCC-3'
$S_{bi1}^{10}$	5'-TCCTAGGACT-3'
$S_{bi2}^{10}$	3'-GTCTGTCAGG-5'
$S_{bo1}^{10}$	3'-GCAATCCTGA-5'
$S_{bo2}^{10}$	5'-CAGACAGTCC-3'

**Table 2.** PTs Sticky Ends

Name	DNA sequence
$S_{ei1}^{00}$	3'-ACAGTCC-5'
$S_{ei2}^{00}$	5'-ATCCTGA-3'
$S_{eo1}^{00}$	5'-TCCATGGACTGT-3'
$S_{eo2}^{00}$	3'-CTGGTTCAGGAT-5'
$S_{ei1}^{11}$	3'-ATCCTGA-5'
$S_{ei2}^{11}$	5'-ACAGTCC-3'
$S_{eo1}^{11}$	5'-TCCATGGACTGT-3'
$S_{eo2}^{11}$	3'-CTGGTTCAGGAT-5'
$S_{ei1}^{01}$	3'-ATCCTGA-5'
$S_{ei2}^{01}$	5'-CAAGTCC-3'
$S_{eo1}^{01}$	5'-CGTTAGGACTTG-3'
$S_{eo2}^{01}$	3'-GTCTGTCAGGAT-5'
$S_{ei1}^{10}$	3'-ATCCTGA-5'
$S_{ei2}^{10}$	5'-CAAGTCC-3'
$S_{eo1}^{10}$	5'-CGTTAGGACTTG-3'
$S_{eo2}^{10}$	3'-GTCTGTCAGGAT-5'

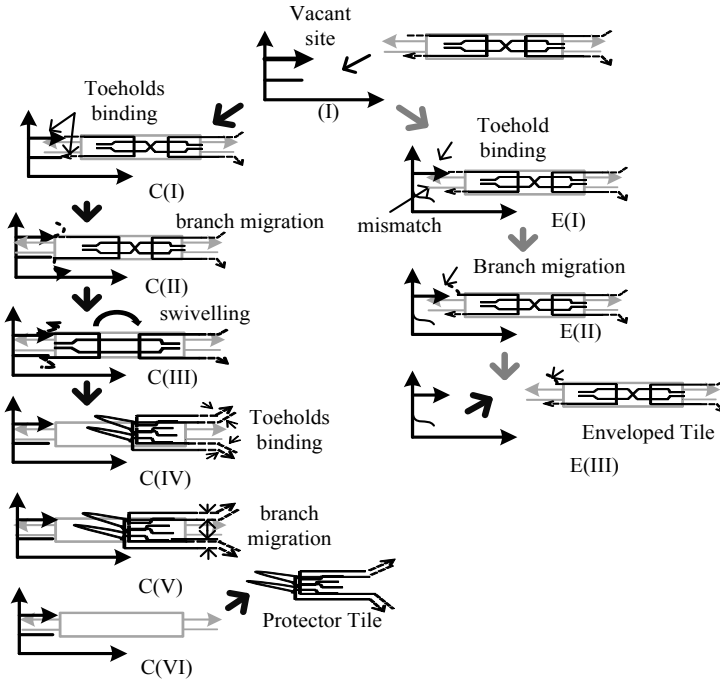
the input and output sequences of the PTs are constrained to have a certain sequence of common bases  $c_1$ ,  $c_2$ , shown in Figure 1(a).

The design of the length of the sticky ends of the *PT* has to take into account a number of factors. These ends must both be able to protect the rule tile's outputs and upon disengaging from the input sides to bond with their input protector counterparts. Therefore, the overhangs have been designed such that the protective helix on the rule tile outputs measures 9 base pairs. Together with the toeholds, the same strands are able to form 7 *nt* bonds internally so as to let the *PT* dissociate from *BT*. The DNA sequences  $c_1$  and  $c_2$  are common for all the *PTs* needed for a Sierpinski tile set. Furthermore, the DNA sequences  $(c_1 + e_{t1}, c_2 + e_{t2})$  are 7 *nt* long, therefore adding 3 bases for protruding toeholds on the output side of *ETs*.

The DNA Design MATLAB scripts [11], were applied to optimise the DNA sequences for *BTs* and corresponding *PTs* for the Sierpinski tile set shown in tables 1 and 2. Due to space constraints, only the sticky end sequences of the *BTs* and *PTs* are given.

## 4 Self-assembly Guided by Enveloped Tiles

As illustrated in Figure 3, when a tile approaches a vacant site (*I*) on the tile assembly, proofreading of the inputs of its *BT* occurs. Such proofreading results in either a match or no match, between the incoming tile and the vacant site. In case of a match, there are 2 potential cases: case 1 — both inputs match correctly (left half of Figure 3) or case 2 — only one input matches correctly (right half of Figure 3).



**Fig. 3.** self-assembly step states: (left) both inputs match, (right) only one input matches

In case 1, the first state – proofreading of the inputs,  $C(I)$  consists of toehold binding followed by branch migration of 7 nt of the inputs of the  $PT$ ,  $C(II)$ , that were bound with a Rule Tile’s sticky ends. A stable binding of the Rule Tile to the assembly has thus occurred. At this stage the Rule Tile no longer needs to be protected by the  $PT$ . As the  $PT$  is released from its input side it will result in uncoiling of the clamp,  $C(III)$ , which primes a configuration in which the left part of the  $PT$  is free to swivel back towards its outputs where it is still protecting the output sticky ends of the  $BT$ . The released inputs start ‘sniffing’ the protruding toeholds at the outputs of the  $PT$ ,  $C(IV)$ . The resulting toehold binding, followed by a further branch migration,  $C(V)$ , of the respective 4 nt will create a single thermodynamic component to disengage the  $PT$  and expose the  $BT$ ’s output sticky ends,  $C(VI)$ . The next incoming, matching Rule Tile, provides additional thermodynamic impetus to further support migration of the  $PT$  away from the growing assembly.

The first stage of Case 2 is as in Case 1, except that only one of the toeholds binds,  $E(I)$ , and initiates branch migration at that input. All other sticky ends of the  $BT$  will remain protected,  $E(II)$ . As this will only occur on one of the input sides all other sticky ends of the non-matching tile will remain protected by the

$PT$  and thus remain unbound to the assembly. Although the branch migration of  $E(II)$ , may or may not proceed to completion, the otherwise fully protected tile will eventually disengage from the lattice due to the inherently unstable nature of binding through a single input,  $E(III)$ . The uniqueness of this approach lies in the strategy applied to protect  $BT$ 's output. The protection comes not just in the form of the binding the outputs but further the distal clamps design protects the outputs from unbinding unless the tile is a correct tile.

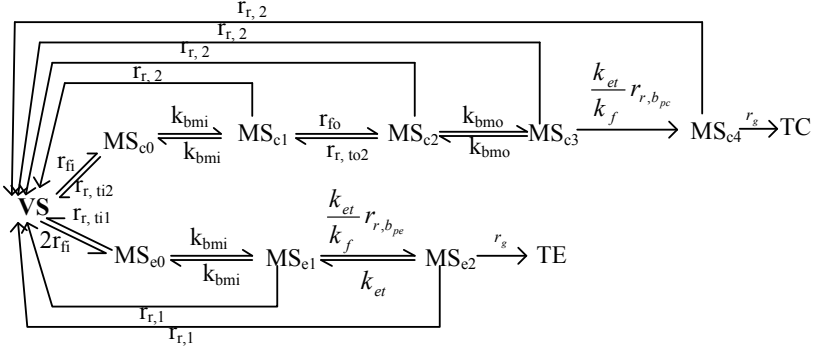
## 5 Performance Analysis

Figure 4 illustrates a continuous time Markov state model of the enveloped tile assembly process. The sequences of states  $(MS_{e0}, MS_{c1}, MS_{c2}, MS_{c3}, MS_{c4})$  and  $(MS_{e0}, MS_{e1}, MS_{e2})$  represent the assembly process of a correct tile and erroneous tile, respectively. These states are equivalent to the assembly steps  $(C(I), C(II), C(IV), C(V), C(VI))$  and  $(E(I), E(II), E(III))$ , respectively, as shown in Figure 3. The states  $TC$  and  $TE$  represent the final states of the corresponding Markov chains where either a correct or an erroneous tile gets trapped.

The state transition  $(VS \rightarrow MS_{c0})$  represents binding between two 3  $nt$  long DNA toeholds of a correctly matching  $ET$  with a vacant site. This has been modelled as a reversible process having forward kinetic rate ( $r_{fi} = k_{fh} e^{-Gmc}$ ) and reverse kinetic rate ( $r_{r,tti2} = k_{fh} e^{-\frac{2 \times 3}{10} Gse}$ ) where  $k_{fh} = 4 \times 10^6 / M / s^2$  [9] is a kinetic rate constant for DNA hybridisation. In the case of  $ET$  having single toehold matching with a vacant site, the state transition  $(VS \rightarrow MS_{e0})$ , two such tiles can engage with the vacant site simultaneously. Thus giving an equivalent forward and reverse rate of  $2r_{fi} = 2k_{fh} e^{-Gmc}$  and  $r_{r,tti1} = k_{fh} e^{-\frac{3}{10} Gse}$ , respectively.

The next step involves branch migration of the  $PT$ 's inputs, where the  $PT$  disengages from the  $BT$ 's input side ( $MS_{c0} \rightarrow MS_{c1}$ ). In the case of single toehold matching, branch migration may still start at this input of  $PT$  but it is not thermodynamically favourable due to the stiffness of the distal clamps reinforcing the  $PT$ 's return to its bind state ( $MS_{e0} \rightarrow MS_{e1}$ ). The aforementioned branch migration step has been modelled as a reversible process [7] where a 'breaking-forming' base pairing process ideally has no preferred direction. The time required is  $\approx 10 \mu S / \text{base-pair}$ , which gives an equivalent kinetic rate  $k_{bmi} \approx 10^3 / s$  in both directions. However, to include a bias in the branch migration processes involving correct and erroneous tiles, two additional dimensionless parameters  $E_{dc}$  and  $E_{de}$  ( $E_{dc} = \frac{E_{dc}}{10}$ ) have been introduced. The new reverse kinetic rates (not shown in Figure 4) are  $k_{bmi} e^{-E_{dc}}$  and  $k_{bmi} e^{E_{de}}$ , reflecting a correct tile and erroneous tile, respectively. Therefore, a larger value of  $E_{dc}$  would facilitate release of  $PT$  from its input side when a correct  $ET$  binds however in the case of erroneous  $ET$  binding it would resist the release of  $PT$  resulting in better error prevention. For simulation purposes,  $E_{dc} = 10$  is chosen.

The next step in the addition of a correct tile ( $MS_{c1} \rightarrow MS_{c2}$ ) represents the binding between the toeholds on the output sides of a  $PT$  and its own input sticky ends that have just been released. As described above, following the release of



**Fig. 4.** Kinetic flow graph of Enveloped Tile assembly

the inputs and prior to engaging with its own output strands, the *PT* undergoes a conformational change by swivelling around its central clamp. It seems likely to assume that this happens instantaneously because the dimensions of a *PT* are in the range of 10 nm. Therefore, this step has been modelled as a reversible process of DNA hybridisation having a forward rate ( $r_{fo} = k_{fh} e^{-G_{mc}}$ ) and a reverse rate ( $r_{r, to2} = k_{fh} e^{-\frac{2 \times 3}{10} G_{se}}$ ).

The following step ( $MS_{c2} \rightarrow MS_{c3}$ ) represents the branch migration of complementary sections following toeholds on the output ends of the *PT*. This has been modelled as an unbiased reversible process with kinetic rates of  $K_{bmo} = 10^3/s$ . Finally, the *PT* dissociates from its *BT* ( $MS_{c3} \rightarrow MS_{c4}$ ), modelled as having a unidirectional kinetic rate of dissociation  $\frac{k_{et}}{k_f} r_{r, b_{pc}}$  where  $r_{r, b_{pc}} = k_{fh} e^{-b_{pc} G_{se}}$  and  $b_{pc} = 1$ . The  $b_{pc}$  represents the total strength with which a *PT* remains still attached with the *BT* after its input ends and output ends bind together. The  $k_{et}$  represents the effective kinetic rate of localisation of the *PT* within a *BT*, which together occupy a volume of  $10^3 \text{ nm}^3$  and therefore, gives an ‘effective concentration’  $C_t = \frac{1}{10^3 \text{ nm}^3} \text{ M}$ . Therefore  $k_{et} = k_{fh} C_t \approx 6 \times 10^4$ . The *PT* dissociation step in the case of an erroneous tile ( $MS_{e1} \rightarrow MS_{e2}$ ) has parameters similar to a correct tile except that  $r_{r, b_{pe}} = k_{fh} e^{-b_{pe} G_{se}}$  and  $b_{pe} = 27/10$ , which would lead to very slow rates of dissociation under given experimental conditions. The kinetic rate  $r_g$  of the final steps ( $MS_{c4} \rightarrow TC$  and  $MS_{e2} \rightarrow TE$ ) represents equivalent kinetic rate of tile trapping.

If  $p_i(t)$  denotes the probability of the aggregate-tile complex being in state ( $i$ ) at a time  $t$  then the self-assembly kinetic rate flow graph of Figure 4 can be written in the form of a Matrix differential equation  $\dot{p}(t) = Mp(t)$  where  $p(t)$  is a  $11 \times 1$  matrix having  $p_i(t)$  ( $i$  represents corresponding Markov state) as its elements and

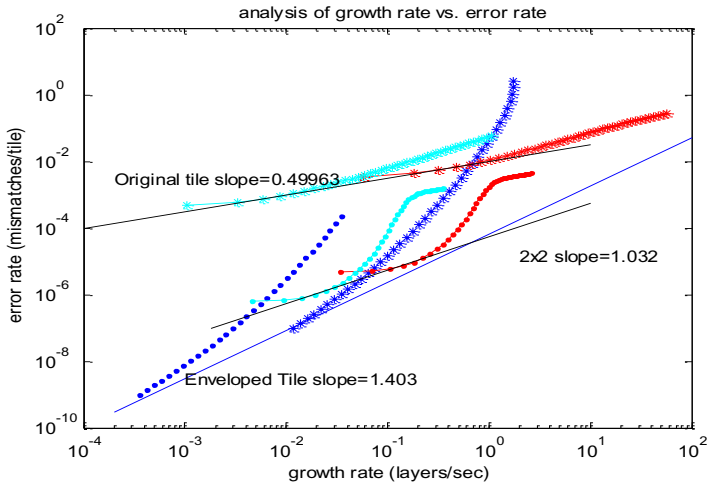
$$M = \begin{bmatrix} -3r_{\beta} & r_{r,ti2} & r_{r,2} & r_{r,2} & r_{r,2} & r_{r,2} & r_{r,1} & r_{r,1} & 0 & 0 & 0 \\ r_{\beta} & -k_{bmi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{bmi} & -(r_{r,2}+k_{bmi}+r_{fo}) & r_{r,ti2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_{fo} & -(r_{r,2}+k_{bmo}+r_{r,ti2}) & -(r_{r,2}+r_i) & k_{bmo} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{bmo} & -(r_{r,2}+k_{bmo}+\frac{k_{et}}{k_f}r_{r,b_{pc}}) & k_{et} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{k_{et}}{k_f}r_{r,b_{pc}} & -(r_{r,2}+k_{et}+r_g) & 0 & 0 & 0 & 0 & 0 \\ 2r_{\beta} & 0 & 0 & 0 & 0 & 0 & -(r_{r,ti1}+k_{bmi}) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & k_{bmi} & -(r_{r,1}+k_{bmi}+\frac{k_{et}}{k_f}r_{r,b_{pc}}) & r_{et} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{k_{et}}{k_f}r_{r,b_{pc}} & -(r_{r,1}+k_{et}+r_g) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_g & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_g & 0 \end{bmatrix}$$

The above matrix differential equation has standard solution as  $p(t) = e^{Mt}p(0)$  where  $p(0) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T$ , and another observation that every tile would get fixed either correctly or erroneously if self-assembly is allowed to continue for longer time, gives  $\dot{p}(\infty) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ p_{TE}(\infty) \ p_{TC}(\infty)]^T$ . The terms  $p_{TC}(\infty)$  and  $p_{TE}(\infty)$  refer to the states  $TC$  and  $TE$ , respectively, when  $(t \rightarrow \infty)$ . The growth rate ( $r_g$ ) can be given by the net transition rate from  $VS$  to  $MS_{c4}$  of the Markov chain shown in Figure 4.

$$r_g = \frac{1}{1/(r_{fi}-r_{r,ti2})+2/(k_{bmi})+1/(k_{dc}-k_{et})+1/(r_{r,fo}-rr_{,to2})} - \frac{r_{r,2}}{4}$$

The error rate vs. growth rate for the enveloped tile assembly has been calculated applying above derivations. Figure 5 provides an analysis of the error rate vs. growth rate for the enveloped tile, the original tile set and the 2x2 proof-reading tile set (data from [10]). The original tile and 2x2 tile are represented by two curves corresponding to  $G_{se} = 6.5, 8.5$  and  $G_{se} = 6.5, 7.5$ , respectively. The  $G_{mc}$  for these plots vary according to the relation ( $G_{mc} = 2 G_{se} - \epsilon$ ) while keeping  $G_{se}$  constant. For the enveloped tile simulations, two curves represent  $G_{se} = 7.5$  and  $G_{se} = 8.5$ , respectively.

The optimum performance (error rate vs. growth rate) of each tile type is achieved along their respective slope  $\frac{d(\log(\text{error rate}))}{d(\log(\text{growth rate}))}$  lines. Further, these slopes may be used to derive relations between growth rate ( $r_g$ ) and error rate ( $\epsilon$ ). For enveloped tile, this relation is  $r_g \approx \beta_{et}\epsilon^{0.7}$  where  $\beta_{et} \approx 8.25 \times 10^4/M/sec$ . Error rate and growth rate relations for original tile, 2x2 tile, and Layered and Protected tiles of [4] are: Original Tile:  $r_g \approx \beta_{original}\epsilon^2$  ( $\beta_{original} \approx 1 \times 10^4/M/sec$ ); 2x2 Tile:  $r_g \approx \beta_{2x2tile}\epsilon^{1.4}$  ( $\beta_{2x2tile} \approx 1 \times 10^4/M/sec$ ); Protected Tile:  $r_g \approx \beta_{PTM}\epsilon^{1.4}$  ( $\beta_{PTM} \approx 4.4 \times 10^2/M/sec$ ); Layered Tile:  $r_g \approx \beta_{LTM}\epsilon^{0.7}$  ( $\beta_{LayeredTile} \approx 3.6 \times 10^2/M/sec$ ). Thus, the enveloped tile are more robust against errors than the original tile and 2x2 tiles whilst comparable to the Layered Tile.



**Fig. 5.** Analytically derived growth rate vs. error rate plots of Original Tile ( $G_{se}=6.5, 8.5$  (top curves)), 2x2 tile ( $G_{se}=6.5, 7.5$  (middle curves)) and Enveloped Tile ( $G_{se}=7.5, 8.5$  (bottom curves))

## 6 Conclusion and Future Work

A new Enveloped Tile approach for reliable tile assembly has been presented, together with an analysis of the involved association and dissociation processes necessary for correct tile assembly. The design of a complete Enveloped Tile set for the Sierpinski Triangle assembly was presented, as an example of an application of the new approach. Error resilience of the enveloped tile assembly has been analysed using a physically realistic kinetic trapping model. Error rate vs. growth rate assessment revealed that the enveloped tile's error prevention capability is equivalent to (if not better than) the best case of the Layered tile.

It is anticipated that the fully protected output of Enveloped Tiles may in addition provide better prevention of other errors, i.e. facet nucleation and spontaneous nucleation errors, which warrants thorough analysis in further work. Other features that remain to be investigated include the impact of coiling, stiffness and swivelling of the central clamp of the *PT* in order to achieve a more accurate modelling of enveloped tile-mediated self-assembly.

The self-triggered activation principle aimed to guide an error-free tile assembly process may provide a general approach to designing molecular systems for DNA computing, synthetic biology and nanotechnology, where different components are often required to co-exist in order to allow conditional interactions and therefore be able to interact also illegitimately.



## References

1. A., W.P., van der, H.T., Fernando, M.H., pakowitz Andrew S, Rob, P., Jonathan, W., Cees, D., Nelson, P.C.: : High flexibility of dna on short length scales probed by atomic force microscopy. *Nature* 2(1), 1748–3387 (2006)
2. Bhalla, N., Bentley, P., Vize, P., Jacob, C.: Staging the self-assembly process using morphological information. In: *European Conf. on Artificial Life (ECAL)*, pp. 93–100 (2011)
3. Chen, H.L., Schulman, R., Goel, A., Winfree, E.: Reducing facet nucleation during algorithmic self-assembly. *Nano Lett.* 7(9), 2913–2919 (2007)
4. Fujibayashi, K., Zhang, D.Y., Winfree, E., Murata, S.: Error suppression mechanisms for dna tile self-assembly and their simulation. *Natural Computing* 8(3), 589–612 (2009)
5. Chen, H.-L., Goel, A.: Error free self-assembly using error prone tiles. In: Ferretti, C., Mauri, G., Zandron, C. (eds.) *DNA10. LNCS*, vol. 3384, pp. 62–75. Springer, Heidelberg (2005)
6. Majumder, U., LaBean, T.H., Reif, J.H.: Activatable tiles: Compact, robust programmable assembly and other applications. In: Garzon, M.H., Yan, H. (eds.) *DNA 13. LNCS*, vol. 4848, pp. 15–25. Springer, Heidelberg (2008)
7. Panyutin, I., Hsieh, P.: The kinetics of spontaneous dna branch migration. *PNAS* 91, 2021–2025 (1994)
8. Schulman, R., Winfree, E.: Programmable control of nucleation for algorithmic self-assembly. In: Ferretti, C., Mauri, G., Zandron, C. (eds.) *DNA 10. LNCS*, vol. 3384, pp. 319–328. Springer, Heidelberg (2005)
9. Winfree, E.: On the computational power of dna annealing and ligation. In: *DNA Based Computers. DIMACS*, vol. 27. American Mathematical Society (1995)
10. Winfree, E., Bekbolatov, R.: Proofreading tile sets: Error correction for algorithmic self-assembly. In: Chen, J., Reif, J.H. (eds.) *DNA 9. LNCS*, vol. 2943, pp. 126–144. Springer, Heidelberg (2004)
11. Winfree, E., Liu, F., Wenzler, L.A., Seeman, N.: Design and self-assembly of two-dimensional dna crystals. *Nature* 394(6693), 539–544 (1998)
12. Woo, S., Rothmund, P.W.K.: Programmable molecular recognition based on the geometry of dna nanostructures. *Nat. Chem.* 3(8), 620–627 (2011)

# Learning from Uncertain Data Using Possibilistic Artificial Immune Recognition Systems

Rim Hentech, Ilyes Jenhani, and Zied Elouedi

LARODEC, Université de Tunis, Institut Supérieur de Gestion de Tunis  
41 Avenue de la liberté, cité Bouchoucha, 2000 Le Bardo, Tunisia  
rimhentech@gmail.com, ilyes.j@lycos.com, zied.elouedi@gmx.fr

**Abstract.** This paper presents a new approach in machine learning, especially, in supervised classification and reasoning under uncertainty. For many classification problems, uncertainty is often inherent in modeling applications and should be treated carefully and not rejected in order to make better decisions. Artificial Immune Recognition System (AIRS) is a well known classifier that has provided good results in the certain context. However, this method is not able to cope with uncertainty. In order to overcome this limitation, we propose a new classification approach combining the AIRS and possibility theory. The new approach is allowing to deal with uncertain attribute values of training instances. The uncertainty is expressed via possibility distributions. Experimentations on real data sets from the U.C.I machine learning repository show good performances of the proposed approach.

**Keywords:** Artificial immune recognition system, Possibility theory, Classification under uncertainty.

## 1 Introduction

Classification techniques are among the well known machine learning methods. They work under a supervised mode where data are labeled with pre-defined classes and new unlabeled data will be classified into these a priori known classes. The artificial immune recognition system (AIRS) has shown to be a good competitor when compared to well established classifiers [13]. AIRS is an immune-inspired supervised learning technique based on clonal selection theory of natural immune systems. It has proven its performance in several areas such as medical diagnosis problems [15, 18, 19], software fault prediction problems [2], intrusion detection problems [19], etc.

Despite the AIRS works very well in an environment characterized by certain and precise data, it does not deal with uncertain data. In fact, uncertainty in data sometimes exists due to noise, residual variation, unreliable sensors, etc. Handling uncertain data using uncertainty theories has attracted a lot of attention. The uncertainty can affect the class labels as well as the attributes values of training objects [1, 7]. Ignoring this uncertainty can influence the results and makes it in some cases, inaccurate. However, dealing with this kind of imperfection can be of great importance and give us information that can help in better

decision making. So, dealing with datasets containing uncertain attribute values is very important due to the fact that this kind of datasets is inevitable in real world data mining applications. Historically, the probabilistic method is considered to be the first framework handling all kinds of imperfection. However, this theory is not an appropriate tool when facing to the case of total ignorance. That is why many other uncertain tools have appeared to overcome this limitation. Among the panoply of theories, possibility theory is an uncertainty theory offering a flexible tool for modeling imperfect information, it has been successfully combined with classification techniques such that the possibilistic decision tree, where authors in [1] handle uncertain attributes values in the training set also in [11] they deal with uncertainty in class values and the naive possibilistic network classifier [7] which deals with imprecise data, etc.

Thus, the aim of this work is to develop a new version of AIRS in an uncertain environment. We combine possibility theory and the artificial immune recognition system classifier to deal with uncertain attribute values. The proposed approach is called the possibilistic AIRS (PAIRS).

This paper is organized as follows. Section 2 describes some basics of the artificial immune recognition system and presents an overview of the research done on AIRS under uncertainty. Section 3 gives necessary background on possibility theory. Then, in Section 4, we detail our approach. Section 5, presents and analyzes experimental results carried out on modified versions of commonly used data sets from the U.C.I repository [14]. Finally, Section 6 concludes the paper.

## 2 The Artificial Immune Recognition System

Artificial Immune Recognition System (AIRS) is a resource limited supervised learning algorithm inspired by immune metaphors [22]. It is composed of four focal steps which are (1) initialization, (2) memory cell identification and ARB generation, (3) competition for resources and a development of a candidate memory cell and lastly (4) memory cell introduction. In this paper, we will focus on the second version of AIRS. The second version AIRS2 has been proposed as a revision of the first one [21]. AIRS2 offers simplicity of implementation, more data reduction and minimizes the processing time. In other words, AIR2 has lower complexity, higher data reduction and it has shown high accuracy results.

### 2.1 AIRS2 Algorithm

In this section, we briefly present the AIRS2 algorithm. We start by the first procedure which represents the learning procedure then we move to the classification procedure which uses the k nearest neighbor approach.

**The Learning Procedure:** This procedure represents the main step in the AIRS algorithm. It aims to produce a reduced data set (memory cell pool MC) on which a k-nearest neighbor classifier (in the classification step) will be based to predict the class label of an unseen instance given values of its attributes. Input

of the learning step is the training set  $T$  where each instance of the training data is considered as an antigen which follows the same representation as an antibody [22].

1. **Initialization Step:** This stage is considered as the pre-processing stage. All continuous attributes in the training set are normalized to ensure that the Euclidean distance (Edist) between two data points lies in the interval  $[0,1]$ . The normalized Euclidean Distance serves also as the affinity measure between two cells.

$$affinity(x, y) = Edist(x, y) = \sqrt{\sum_{i=1}^m (x_i - y_i)^2} \quad (1)$$

Where  $x$  and  $y$  are the two attribute vectors representing two instances (cells) and  $m$  represents the number of attributes in the training set.

After the normalization of continuous attributes, the affinity threshold which represents the mean affinity between all antigens in the training set should be calculated according to the following equation:

$$affinity\_threshold = \frac{\sum_{i=1}^n \sum_{j=i+1}^n affinity(ag_i, ag_j)}{\frac{n*(n-1)}{2}} \quad (2)$$

Where  $n$  is the number of antigens in the training data.  $ag_i$  and  $ag_j$  are the  $i^{th}$  and  $j^{th}$  training antigens, and  $affinity(ag_i, ag_j)$  returns the Euclidean distance (or any other adequate distance) between the two antigens' attribute vectors. The value of  $affinity\_threshold$  will be used in next stages.

The final step is the initialization of the memory cell pool (MC). This is done by randomly choosing 0 or more antigens from the set of training instances to be added to the set of memory cells.

2. **Memory Cell Identification and ARBs Generation:**

The AIRS algorithm is a single-shot algorithm that is, only one antigen goes through the training process at a time [23]. In this stage, the algorithm begins to iterate for each training antigen.

- *Clonal Expansion:* For each element  $mc$  of MC, determine its affinity to the antigenic pattern  $ag$ , which has the same class ( $MC_{ag.c}$ ) ( $Stim(ag, mc) = 1 - affinity(ag, mc)$ ). The memory cell ( $mc$ ) with the highest stimulation is then selected as the best match memory cell ( $mc\_match$ ) that will be used in the affinity maturation process. If there is no  $mc$  in the memory cell pool having the same class as the training antigen, then this antigen will be added to the set of memory cells.
- *Affinity Maturation:* The selected memory cell ( $mc\_match$ ) as well as a number of mutated clones of this latter are created according to mutation process and added to the ARB pool AB which contains the mutated clones of  $mc\_match$ . The number of clones is computed by:

$$(mc\_match.Num\_Clones =$$

$$ClonalRate * HyperClonalRate * Stim(ag, mc\_match))$$

where *stim* is the calculated stimulation value and *ClonalRate* and *HyperClonalRate* are user-defined parameters.

The mutation process is done randomly. A random value, between 0 and 1, will be generated for each feature value. If this value is lower than the *MutationRate* (a pre-defined parameter) then mutation will take place. Only the attribute values are allowed to be mutated.

3. **Competition for Resources and Development of a Candidate Memory Cell Step:** At this level, the ARB pool (AB) contains *mc\_match* and mutated clones of *mc\_match*. The purpose of this stage is to develop a candidate memory cell (*mc\_candidate*) which better recognizes the actual antigen *ag*. This is ensured by the following steps:

- Each ARB in the population AB is presented with the antigen *ag* to determine the ARB's stimulation level. This stimulation is then normalized.
- Each ARB in AB is allocated a finite number of resources using the following formula:

$$(ARB.resources = ClonalRate * Stim\_normalized(ag, ARB))$$

- When the total number of resources allocated to all the ARBs exceeds the allowed limit (a user defined-parameter), the surplus resources are removed from the weakest ARBs until the number of resources is equal to the allowed number of resources. Then ARBs left with zero resources will be removed from the system. Once resources are allocated to the different ARBs, we proceed to test if the stopping criterion is met. That is we must test if the average stimulation value for all the existing ARBs with the antigen exceeds a user predefined parameter namely *stimulation\_threshold*. The stopping criterion is met when the average stimulation value for all ARBs exceeds *Stimulation\_threshold*.

If the average stimulation value is lower than the *stimulation\_threshold*, the ARBs will be mutated and the new mutated clones will be added to the ARB pool. Survived ARBs produce new ARBs through clonal expansion and mutation until average stimulation of all classes exceed the *stimulation\_threshold*. Here the number of clones is computed by:  $(ARB.Num\_Clones = Stim(ag, ARB) * ClonalRate)$

- *Mc\_candidate Identification:*

After the total stimulation value of ARBs reaches stimulation threshold, the best ARB is taken as a candidate memory cell (*mc\_candidate*). Here best means having the highest stimulation.

4. **Memory Cell Introduction Step:** This step consists in updating the MC pool. That is, the selected *mc\_candidate* will be added to the MC pool and will become a long-lived memory cell if its affinity for the training antigen

is greater than the affinity of the original memory cell (`mc_match`). Furthermore, if the affinity between the `mc_candidate` and the `mc_match` is less than the product of `Affinity_threshold` and `Affinity_threshold_scalar` (where `Affinity_threshold` is the affinity threshold calculated in the initialization stage and `Affinity_threshold_scalar` is a user defined parameter) than the original memory cell `mc_match` will be removed from the memory cell pool MC.

The ARB pool is cleared after each new antigenic presentation.

The algorithm continues until all the training antigens have been presented and trained.

**Classification Procedure:** When the training process is completed, the memory cell pool (MC) becomes the core of the AIRS classifier. The classification is performed using the k-nearest neighbors (K-nn) approach, the classification of a test data is accomplished by majority vote of the k nearest memory cells to the presented test antigen to be classified.

## 2.2 Background Research on AIRS Under Uncertainty

Many researches have been conducted to improve the accuracy of AIRS and to identify the significant components of AIRS that could empower it for better performance. Some of these researches have been done to adapt AIRS, in an environment characterized by uncertainty. In classical AIRS, resource allocation is done linearly with affinities. This linearity requires excess resource usage in the system, which results in long classification time and high number of memory cells. That is why authors in [15] proposed to replace the linear resource allocation by fuzzy resource allocation in order to increase its classification performance in terms of classification time and number of memory cells. The effect of the fuzzy resource allocation method on the accuracy of AIRS was evaluated by Golzari et al. [5] from a statistical point of view. They found that the fuzzy allocation increases the accuracy of AIRS in majority of data sets. However, the increase is significant in minority of data set.

The same authors incorporated an adapted real world tournament selection method into the resource competition phase of AIRS [6]. Also, Polat and his coauthor in [16] proposed before the application of AIRS, a fuzzy weighted preprocessing procedure based on sample means of each feature values. Two membership functions are defined known as input and output membership functions which are allowed to assign a new feature value for each feature according to its old value. In [17], authors proposed a new weighting preprocessing procedure based on the k nearest neighbor where each attribute value is replaced by the mean of the k-nearest attribute values to the initial attribute among all sample data which improve the classification results.

Most recently in [3], a fuzzy k nearest neighbor was combined with AIRS2 in order to identify the diabetes diseases. The use of fuzzy K-NN increased the accuracy of the classifier. In [19], a rough set theory was coupled with AIRS.

This combination was applied to reduce false alarm rate in the intrusion detection problem.

The existing works and available approaches of AIRS have not explicitly dealt with the uncertainty of the input data. They deal with training data sets containing only certain information. However, data in many real-world applications may be uncertain.

### 3 Possibility Theory: An Overview

Possibility theory is a simple mathematical tool offering a natural and simple model to deal with incomplete information. It is introduced at first by Zadeh [24] and developed later by several researchers mainly Dubois and Prade [4]. In this section, we will give a brief recalling on possibility theory.

#### 3.1 Possibility Distribution

Given a universe of discourse  $\Omega = \{\omega_1, \omega_2, \dots, \omega_n\}$ , a possibility distribution is considered as one of the fundamental concepts of possibility theory. It is denoted by  $\pi$  and it assigns to each element  $\omega_i$  of the universe of discourse  $\Omega$  a value from a bounded linearly ordered scale  $L$ . This value is called possibility degree: it encodes our knowledge on the real world. The possibilistic scale  $L$  can be defined by  $[0,1]$  in the numerical scale where values of the possibility distribution make sense.

By convention  $\pi(\omega_i) = 0$  means that  $\omega_i$  is cannot be the real world,  $\pi(\omega_i) = 1$  means that it is fully possible that  $\omega_i$  is the real world. Moreover, a possibility distribution is said to be normalized if there exists at least one state  $\omega_i$  which is totally possible ( $\max \pi(\omega_i) = 1$ ). In this paper, we only deal with normalized possibility distributions.

Given a possibility distribution  $\pi$ , extreme cases of knowledge can be defined as the complete knowledge ( $\exists \omega_0, \pi(\omega_0) = 1$  and  $\pi(\omega) = 0 \forall \omega \neq \omega_0$ ) and the total ignorance ( $\forall \omega \in \Omega, \pi(\omega) = 1$ ).

#### 3.2 Similarity and Dissimilarity Measures in Possibility Theory

The choice of the similarity measure is dependent on the representation of objects (single value, vector of individual values, sets of values, sets of vectors, etc.) in addition to the domain value of that objects (real values, symbolic values, strings, etc.). In possibility theory, possibilistic similarity measures aim at computing the similarity (or dissimilarity) degree between two possibility distributions. Possibilistic similarity measures are characterized by several properties [12]. Three well-known similarity measures are the most utilized namely the information closeness [8] ( $G(\pi_1, \pi_2) = g(\pi_1, \pi_1 \cup \pi_2) + g(\pi_2, \pi_1 \cup \pi_2)$  where  $g$  is the difference between the uncertainty of two possibility distributions)<sup>1</sup>, Sanguesa

<sup>1</sup>  $g(\pi_i, \pi_j) = |U(\pi_j) - U(\pi_i)|$ .  $\cup$  is taken as the maximum operator and  $U(\pi) = \sum_{i=1}^n [(\pi(\omega_{(i)}) - \pi(\omega_{(i+1)})) \log_2 i] + (1 - \pi_{(1)}) \log_2 n$  corresponds to the non specificity measure where  $\pi(\omega_{(n+1)}) = 0$  and  $n = |\Omega|$ .

et al. distance [20] ( $\text{distance}(\pi_1, \pi_2) = U(|\pi_1 - \pi_2|)$ ) and the information affinity [9, 10, 12] ( $\text{Aff}(\pi_1, \pi_2) = 1 - [D(\pi_1, \pi_2) + \text{Inc}(\pi_1, \pi_2)/2]^2$ ).

## 4 The Possibilistic AIRS Method

As mentioned previously, AIRS2 is considered as one of the most powerful techniques of machine learning due to its efficiency in the classification area. It is widely applied to a variety of problems in artificial intelligence. AIRS2 has shown high accuracy when we deal with training sets characterized by certain training objects where their attributes values are supposed to be known with certainty.

Uncertainty in data always exists owing to intentionally errors and lack of information. Handling data sets containing uncertain attribute values has a great importance due to its existence in several data mining fields. Ignoring this uncertainty could have a negative impact on the result and makes it in some situations erroneous.

From these issues, we develop a possibilistic AIRS method, a new classification technique based on the AIRS method within the possibility theory for handling uncertain attribute values. Our proposed approach aims at learning objects with either certain or uncertain attribute values. In our proposed possibilistic method, we only deal with objects described by categorical attributes.

### 4.1 Notations

We define the notations that will be used in our work:

- $T$  is the set of  $n$  objects such that  $T=(X_1, X_2, \dots, X_n)$ .
- $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$  is the object described by  $m$  categorical attributes.
- $A_j = (a_{j1}, a_{j2}, \dots, a_{jt})$  is the set of  $t$  values of the attribute  $j$  with  $1 \leq j \leq m$ .
- $\pi_{ij}$  is the possibility distribution defined for the attribute  $A_j$  related to the object  $X_i$ .

We will use also the notations provided in Section 2 and Section 3.

### 4.2 The Possibilistic AIRS Algorithm

To work under uncertain framework using possibility theory, our method requires the definition of new parameters based on the new structure of the training set, the computation of the possibilistic similarity measure between an antigen and an antibody and the mutation process.

**The Structure of the Training Set:** In our possibilistic AIRS, we will deal with training antigens whose attributes are described by a possibility distribution. More formally, a possibility degree will be assigned to each possible attribute

---

<sup>2</sup>  $D(\pi_1, \pi_2) = \frac{1}{n} * \sum_{i=1}^n |\pi_1(\omega_i) - \pi_2(\omega_i)|$  ;  $\forall \omega_i \in \Omega = \{\omega_1, \omega_2, \dots, \omega_n\}$  and  $n = |\Omega|$ .  
 $\text{Inc}(\pi_1, \pi_2) = 1 - \max_{\omega} (\pi_1(\omega) \cap \pi_2(\omega))$  with  $\cap$  is chosen as the minimum operator.



value. These possibility degrees generally given by an expert (or a sensor), represent the opinions of this expert about the real values of the attributes for each object in the training set.

In fact, the new structure of the training set represents also special cases of total ignorance and complete knowledge [4].

**The Possibilistic Similarity Measure:** We need to use a possibilistic similarity measure suitable to the possibilistic AIRS method in order to handle uncertain attribute values. The similarity measure can influence the classification result. That is why choosing the appropriate measure which computes efficiently the similarity between two uncertain tuples, is fundamental. The selection of such similarity measure is based on its capacity to satisfy some properties which are mentioned in [12]. The information affinity satisfies all basic properties, it is expected to give the most accurate results, for these reasons we find it as the most adequate to our possibilistic AIRS approach which represents both the stimulation and the affinity between an antigen and an antibody. To determine the similarity degree between an antigen  $ag_1$  and an antibody  $ab$ , we have to define the affinity using this formula:

$$affinity(ag_1, ab) = \frac{\sum_{j=1}^m Aff(\pi_{1j}, \pi_{2j})}{m} \quad (3)$$

where  $m$  corresponds to the number of attributes,  $\pi_{1j}$  the possibility distribution of the attribute  $j$  of  $ag_1$  and  $\pi_{2j}$  the possibility distribution of the attribute  $j$  of  $ab$ . The less the value  $Aff$  is, the more the information are dissimilar.

**The Mutation Process:** The new mutated clones have an impact on the choice of the *mc\_candidate* and consequently on the final result of the AIRS algorithm (in certain or uncertain environment). In order to obtain a good result, we focus on the good generation of new clones.

In our mutation algorithm, we will use the notion of wrapper possibility distribution which is defined in [12] as follows:

Wrapper possibility distributions are binary possibility distributions (i.e.,  $\forall \omega \in \Omega, \pi(\omega) \in \{0, 1\}$ ) representing special cases of complete knowledge, partial ignorance and total ignorance representing the set of reference distributions.

Given the training set  $T$  containing  $n$  instances and given the set of attributes, let us denote by  $\pi_{ij}$  the possibility distribution that defines the attribute  $A_j$  of the instance  $X_i$  in  $T$  and  $t$  represents the number of values of the attribute  $A_j$ . We define a degree of imprecision  $dimp \in \{1, \dots, t\}$  which will allow us to determine the set of WD to be used.

Here, we suppose that we are dealing with an attribute with  $t=3$ . The set of wrapper distributions relative to this attribute is defined as follows:

◆ If  $dimp = 1$  (First-Level Wrapper Distributions), we will only consider wrapper possibility distribution corresponding to complete knowledge, i.e.,  $WD = [1, 0, 0], [0, 1, 0], [0, 0, 1]$ .

◆ If  $\text{dimp} = 2$  (Second-Level Wrapper Distributions), we will also consider partial ignorance, i.e.,  $\text{WD} = [1, 0, 0], [0, 1, 0], [0, 0, 1], [1, 1, 0], [1, 0, 1], [0, 1, 1]$ .

◆ If  $\text{dimp} = 3$  (Third-Level Wrapper Distributions), we will also consider total ignorance, i.e.,  $\text{WD} = [1, 0, 0], [0, 1, 0], [0, 0, 1], [1, 1, 0], [1, 0, 1], [0, 1, 1], [1, 1, 1]$ .

After specifying the set of WD, relative to this attribute, we will assign the most similar wrapper distribution  $WD_k$  such that

$$k = \underset{k=1}{\operatorname{argmax}} \{ \text{Aff}(\pi_{ij}, WD_k) \}$$

to this attribute, where  $n_a$  is the total number of wrapper distributions and  $\text{Aff}$  corresponds to the information affinity. During this phase, new clones will be generated by the mutation process. This process ensures the diversification of the population. Therefore, generating new ARB may help to obtain the  $mc\_candidate$  which is more similar to the antigen than  $mc\_match$ . Our aim is to increase the affinity between the antigen and the clones of  $mc\_match$ . In order to adapt this process to our possibilistic framework, the wrapper distribution is defined by fixing the most stimulated wrapper to reach. This mechanism allows us to obtain good clones which affect the final results.

The mutation process is reported as follows: first, we calculate for each  $\pi_{ij}$  of the training antigen the affinity with the wrapper distributions. Then, the most similar wrapper distribution will be selected and denoted by  $WD_{rep}$ . After that, we mutate  $\pi_{ij}$  of the  $mc\_match$ , and then new distribution of this feature will be generated.

The similarity between the new distribution and the wrapper distributions is computed. The most similar wrapper possibility  $WD_{pd}$  will be chosen. If the two selected wrapper distributions are equal, then we move to the next feature and perform the same process. Otherwise, we repeat the mutation process until the stopping criterion is met.

## 5 Experimental Results

### 5.1 The Framework

We have performed several tests and simulations on possibilistic versions of real datasets obtained from the U.C.I. repository [14]. We have used seven data sets namely Hayes-roth (H), Congression voting records (V), Balance-scale (B), Tic-Tac-Toe Endgramme (T), Solar-Flar (S), Car Evaluation (C) and Nursery (N).

### 5.2 Artificial Uncertainty Creation in the Training Set

Possibility degrees are created artificially. The pre-treatment procedure of the UCI databases is described as follows:

**Algorithm 1.** Possibilistic AIRS Mutation Algorithm

---

```

Data: ab, ag, Mutation_Rate, mc_match.Num_Clones , m, num_itr
Result: AB
1 begin
  /* Initialization */
2 mutate(ab) ← false ;
3 AB ← ∅ ;
4 for  $j=1$  to mc_match.Num_Clones do
5   for  $i=1$  to m do
6     /* ab.f(i) represents the attribute i of the antibody ab */
7     ab.f(i).change ← drandom;
8     if  $drandom < mutation\_Rate$  then
9       max ← 0;
10      count ← 0;
11      rep ← 0;
12      pd ← 0;
13      repeat
14        for  $k=1$  to t do
15          /* attribute random values to each possibility
16             degree of the feature i */
17          ab.f(i). $\pi_{ji}(k)$  ← drandom1;
18          if  $max < ab.f(i).\pi_{ji}(k)$  then
19            max ← ab.f(i). $\pi_{ji}(k)$ ;
20          for  $k=1$  to t do
21            ab.f(i). $\pi_{ji}(k)$  ← (ab.f(i). $\pi_{ji}(k)$ )/max;
22          /* Select the  $WD_l$  having the highest affinity with
23             ag.f(i) */
24          rep = arg max $_{l=1}^{na}$  Aff (ag.f(i). $\pi_{ji}$ ,  $WD_l$ );
25          pd = arg max $_{l=1}^{na}$  Aff (ab.f(i). $\pi_{ji}$ ,  $WD_l$ );
26          if  $WD_{rep} = WD_{pd}$  then
27            mutate(ab) ← true ;
28          count ← count + 1 ;
29      until (( $WD_{rep} = WD_{pd}$ ) or ( $count > num\_itr$ ));
30   AB.add(ab);

```

---

- In the case of certain attribute values, a possibility degree 1 is assigned to the true attribute value  $a_{jt}$ . However, we assign a possibility degree 0 to the remaining attribute values ( $\langle \rangle a_{jt}$ ).
- In the case of uncertain attribute values, the true attribute value  $a_{jt}$  takes the possibility degree 1. Then, we allow random possibility degrees from  $[0, 1]$  to the remaining attribute values ( $\langle \rangle a_{jt}$ ).

### 5.3 Evaluation Criteria

The evaluation of our possibilistic AIRS classifier was performed based mainly on three evaluation criteria, namely, the classification accuracy, expressed by the percentage of correct classification (PCC), the running time (t) for training phase and the percentage of data reduction (reduct) which provides the percentage of reduced data among the total number of training instances (a summarized version of the original data set that can be used for the classification step).

In our simulations, in order to obtain an unbiased estimation of the PCC, we have used a certain number of tests and after that we will calculate the final PCC as the average of all obtained ones. This strategy is called cross validation.

**Results for the Certain Case:** In the following, we compare the PCC, the execution time and the percentage of data reduction criteria of AIRS with PAIRS based on the different similarity measures (the information affinity ( $PAIRS_{Aff}$ ), the information closeness ( $PAIRS_{infoclos}$ ) and Sanguesa et al. distance ( $PAIRS_{sang}$ )).

**Table 1.** The PCC (%) of the AIRS2 vs PAIRS with k=1

Databases	H	V	B	T	S	C	N
AIRS	66.2	<b>92</b>	75.64	51.3	80.11	66.8	57.9
$PAIRS_{Aff}$	<b>79.1</b>	79.2	<b>78.71</b>	<b>63.9</b>	<b>81.3</b>	<b>74.8</b>	<b>62.4</b>
$PAIRS_{infoclos}$	76	70.3	76.02	57.8	80.71	69.2	59.1
$PAIRS_{sang}$	78.2	76.1	76.5	60	83.1	70.9	60.2

**Table 2.** The PCC (%) of the AIRS2 vs PAIRS with k=5

Databases	H	V	B	T	S	C	N
AIRS	62.8	<b>93.3</b>	72.45	56.7	74.34	70.2	62.2
$PAIRS_{Aff}$	<b>76.7</b>	82.3	<b>76.12</b>	<b>59.9</b>	<b>83.7</b>	<b>79.6</b>	<b>70.1</b>
$PAIRS_{infoclos}$	71.5	77.6	74.15	57.7	76.45	71.1	62.9
$PAIRS_{sang}$	74.2	81.2	75.97	57.9	79.23	76.8	66.9

Looking at Table 1 and Table 2, generally, the PAIRS outperforms AIRS. For example for the dataset S with k=5, the PCC reaches 74.34% with AIRS however

it reaches respectively 83.7% with  $PAIRS_{Aff}$ , 76.45% with  $PAIRS_{infoclos}$  and 79.23% with  $PAIRS_{sang}$ . Only for the database V, AIRS gives a better accuracy than our possibilistic approach. For example for  $k=5$ , the PCC of the dataset V reaches 93.3% with AIRS however it reaches 82.3% with the  $PAIRS_{Aff}$ , 77.6% with the  $PAIRS_{infoclos}$ , 81.2% with the  $PAIRS_{sang}$ . We can note that, for this database, the simple matching measure leads to obtain better results.

For both  $k=1$  and  $k=5$ , the  $PAIRS_{Aff}$  gives better results in term of PCC.

**Table 3.** The running time (seconds) and the percentage of data reduction of the AIRS2 vs PAIRS

Databases	H	V	B	T	S	C	N
<b>AIRS</b>							
t (s)	<b>0.752</b>	<b>27.45</b>	38.22	<b>44.17</b>	<b>98.45</b>	173.4	<b>5125</b>
Reduct (%)	42.21	<b>50.08</b>	55.06	5.11	40.6	35.76	78.22
<b>PAIRS<sub>Aff</sub></b>							
t (s)	0.835	39.77	<b>32.55</b>	51.20	125.17	<b>163.5</b>	5314
Reduct (%)	<b>44.5</b>	49.89	<b>58.4</b>	<b>7.34</b>	<b>42.87</b>	<b>40.38</b>	<b>81.95</b>
<b>PAIRS<sub>infoclos</sub></b>							
t (s)	0.849	44.674	37.14	60.45	136.72	171.9	5341.5
Reduct (%)	37.43	43.71	49.5	4.69	35.3	31.82	72.55
<b>PAIRS<sub>sang</sub></b>							
t (s)	0.841	40.84	35.75	54.70	132.48	166.22	5327.89
Reduct (%)	40.22	47.88	52.9	5.32	37.8	35.70	75.67

Table 3 illustrates the execution time and the percentage of data reduction of the AIRS2 vs PAIRS. From Table 3, in some datasets the execution time in AIRS is lower than the PAIRS. For datasets having many attributes, the new approach needs more time due to the use of the wrapper possibility distribution in the mutation algorithm. For example, for the dataset Hayes which has 54 attributes, the execution time reaches 0.752 with AIRS. However, with  $PAIRS_{Aff}$  the running time is 0.835 and respectively 0.849 and 0.841 with  $PAIRS_{infoclos}$  and  $PAIRS_{sang}$ .

In terms of execution time, the PCC and the percentage of data reduction, we can note that  $PAIRS_{Aff}$  outperforms the  $PAIRS_{infoclos}$ ,  $PAIRS_{sang}$  and AIRS.

**Results for the Uncertain Case:** Let  $P$  be the percentage of uncertain attributes and  $g$  is the range of the possibility degree defined for the different values of a given attribute.  $P$  and  $g$  are useful to test the behavior of our classifier and its robustness in dealing with many uncertain attributes with different range of possibility degree.

Tables 4 and 5 present a comparison between the PCC of the three possibilistic approaches ( $AIRS_{Aff}$ ,  $AIRS_{infoclos}$  and  $AIRS_{sang}$ ) based on different values of both  $P$  and  $g$ .

**Table 4.** The PCC (%) of the PAIRS with  $k=1$ 

Databases	H	V	B	T	S	C	N
$P < 50\%$							
$0 \leq g < 0.5$							
PAIRS <sub>Aff</sub>	<b>81.3</b>	<b>84.2</b>	<b>77.15</b>	<b>58.9</b>	<b>80.2</b>	<b>78.1</b>	<b>72.1</b>
PAIRS <sub>infoclos</sub>	66.3	72.3	72.2	54	73.5	69.5	65.5
PAIRS <sub>sang</sub>	75.1	74.21	74.21	54.2	77.71	74.1	69.6
$0.5 \leq g \leq 1$							
PAIRS <sub>Aff</sub>	<b>78.2</b>	<b>82.1</b>	<b>82.5</b>	<b>57</b>	<b>83.2</b>	<b>80.3</b>	<b>73.2</b>
PAIRS <sub>infoclos</sub>	69.9	75.1	73.9	51.3	76.81	70.3	69.4
PAIRS <sub>sang</sub>	74.1	79.5	76.21	55.2	79.6	76.4	71.5
$P \geq 50\%$							
$0 \leq g < 0.5$							
PAIRS <sub>Aff</sub>	<b>83.3</b>	<b>82.3</b>	<b>81.01</b>	<b>57.1</b>	<b>83.3</b>	<b>81.9</b>	<b>76.4</b>
PAIRS <sub>infoclos</sub>	75.5	76.5	79.43	50.7	75.30	72.1	70.9
PAIRS <sub>sang</sub>	79.9	80.1	80.5	54.8	81.76	77.2	73.2
$0.5 \leq g \leq 1$							
PAIRS <sub>Aff</sub>	<b>83.4</b>	<b>86.5</b>	<b>84.67</b>	<b>60.3</b>	<b>87.29</b>	<b>84.1</b>	<b>78.4</b>
PAIRS <sub>infoclos</sub>	78.5	79.5	78.3	54.1	76.30	77.3	71.1
PAIRS <sub>sang</sub>	81.2	83.1	81.5	58.2	78.34	83.9	74.6

**Table 5.** The PCC (%) of the PAIRS with  $k=5$ 

Databases	H	V	B	T	S	C	N
$P < 50\%$							
$0 < g < 0.5$							
PAIRS <sub>Aff</sub>	<b>80.3</b>	<b>78.9</b>	<b>80.9</b>	<b>55.4</b>	<b>80.14</b>	<b>77.1</b>	<b>71.4</b>
PAIRS <sub>infoclos</sub>	75.3	74.5	69.56	50	74.12	66.3	65.5
PAIRS <sub>sang</sub>	78.2	78.13	75.37	52.3	77.63	74.9	70.1
$0.5 \leq g \leq 1$							
PAIRS <sub>Aff</sub>	<b>83.3</b>	<b>82.9</b>	<b>81.3</b>	<b>59.4</b>	<b>84.21</b>	<b>76.1</b>	<b>69.1</b>
PAIRS <sub>infoclos</sub>	70.3	76.5	76.62	52.9	79.64	70.1	62.5
PAIRS <sub>sang</sub>	79.2	81.1	80.3	57.3	83.17	75.3	67
$P \geq 50\%$							
$0 < g < 0.5$							
PAIRS <sub>Aff</sub>	<b>81.9</b>	<b>84.3</b>	<b>79.63</b>	<b>53.4</b>	<b>81.63</b>	<b>78</b>	<b>75.1</b>
PAIRS <sub>infoclos</sub>	77.9	80.5	72.67	50	76.34	70.5	65.2
PAIRS <sub>sang</sub>	79.3	81.5	75.23	53.2	79.4	76.4	71.1
$0.5 \leq g \leq 1$							
PAIRS <sub>Aff</sub>	<b>84.3</b>	<b>85.7</b>	<b>83.67</b>	<b>56.5</b>	<b>86.52</b>	<b>85.3</b>	<b>78.2</b>
PAIRS <sub>infoclos</sub>	78.1	78.3	77	50.9	78.41	77.5	68.5
PAIRS <sub>sang</sub>	80.3	81.1	81.96	54.5	82.16	82.3	71.3

From Tables 4 and 5, we can conclude that, for the uncertain case with  $k=5$  and  $P \geq 50\%$ , the difference between the three versions of PAIRS, in term of PCC, is noticeable when  $0.5 \leq g \leq 1$ . The PCC of the  $PAIRS_{Aff}$  is higher than that of both the  $PAIRS_{infoclos}$  and the  $PAIRS_{sang}$ .

In the most datasets, from all listed tables, we can see that our approach achieves encouraging percentage of PCC when  $P > 0.5$  and  $g \in [0.5, 1]$ . It is clear that even the degree of uncertainty is very high, our approach achieves accurate results. These encouraging results are improved by the use of the information affinity as a similarity measure which deals efficiently with uncertainty.

**Table 6.** The running time of the PAIRS (seconds)

Databases	H	V	B	T	S	C	N
$PAIRS_{Aff}$	<b>0.543</b>	<b>26.213</b>	<b>45.36</b>	<b>51.867</b>	<b>116.45</b>	<b>165.863</b>	<b>4276.12</b>
$PAIRS_{infoclos}$	0.641	39.674	54.6	58.423	125.8	173.135	4365.9
$PAIRS_{sang}$	0.591	35.137	49.12	53.274	119.6	169.33	4310.65

Table 6 illustrates the comparison between the three versions of PAIRS in term of processing time.

We note, for example for the database V, using the information affinity measure, the running time is 26.213s. However, by the use of the sanguesa et al. distance, the processing time is 35.137s. It is clearly that, for all the datasets, the  $PAIRS_{Aff}$  outperforms both the  $PAIRS_{infoclos}$  and the  $PAIRS_{sang}$  due to the use of the manhattan distance and the inconsistency measure which produce results very fast.

**Table 7.** The pourcentage of data reduction of the PAIRS

Databases	H	V	B	T	S	C	N
$PAIRS_{Aff}$	<b>48.34</b>	<b>55.12</b>	<b>59.43</b>	<b>9.87</b>	<b>48.10</b>	<b>45.19</b>	<b>77.45</b>
$PAIRS_{infoclos}$	41.7	49.99	52.7	6.31	37.89	38.76	68.93
$PAIRS_{sang}$	43.71	51.14	53.67	7.46	42.14	42.85	73.03

Furthermore, the percentages of data reduction, shown in Table 7, confirm that using the information affinity measure may leads to a more reduced set of memory cells. It is due to its effectiveness in measuring similarity between an antigen and an antibody.

## 6 Conclusion

This paper presents a new classification approach adapted to an uncertain framework called the possibilistic AIRS method. This approach is based on the AIRS2

method within the possibility theory to deal with uncertainty existing in the training set, especially in instances' attribute values. Based on the PCC, the execution time and the percentage of data reduction, PAIRS performs better than AIRS2 for most datasets in the certain case and our PAIRS gives important results in the uncertain context for different degrees of uncertainty and for all datasets. Moreover, we have tested our PAIRS with three different measures of similarity; it has shown that PAIRS gives better results by applying the information affinity as a similarity measure than using the information closeness and Sanguesa et al. distance.

Finally, some future works have to be mentioned. Since PAIRS deals only with categorical attributes, we aim at extending the application of this algorithm with mixed numeric and categorical attributes. Moreover, we can extend our approach in order to handle the uncertainty in both attribute values and class values which are always an open line of research.

## References

1. Amor, N.B., Benferhat, S., Elouedi, Z.: Qualitative classifications with possibilistic decision trees (2004)
2. Catal, C., Diri, B.: Software defect prediction using artificial immune recognition system. In: Proceedings of the 25th Conference on IASTED International Multi-Conference: Software Engineering, pp. 285–290. ACTA Press, Anaheim (2007)
3. Chikh, M.A., Saidi, M., Settouti, N.: Diagnosis of diabetes diseases using an artificial immune recognition system2 (airs2) with fuzzy k-nearest neighbor. *J. Med. Syst.* 36(5), 2721–2729 (2012)
4. Dubois, D., Prade, H.: Possibility Theory: An Approach to Computerized Processing of Uncertainty (traduction revue et augmentée de "Théorie des Possibilités"). Plenum Press, New York (1988)
5. Golzari, S., Doraisamy, S., Sulaiman, M.N., Udzir, N.I.: Effect of fuzzy resource allocation method on airs classifier accuracy. *Journal of Theoretical and Applied Information Technology* 5, 18–24 (2005)
6. Golzari, S., Doraisamy, S., Sulaiman, M., Udzir, N.: A review on concepts, algorithms and recognition based applications of artificial immune system. In: Sarbazi-Azad, H., Parhami, B., Miremadi, S.-G., Hessabi, S. (eds.) *Advances in Computer Science and Engineering*. CCIS, vol. 6, pp. 569–576. Springer, Heidelberg (2009), [http://dx.doi.org/10.1007/978-3-540-89985-3\\_70](http://dx.doi.org/10.1007/978-3-540-89985-3_70)
7. Haouari, B., Amor, N.B., Elouedi, Z., Mellouli, K.: Naive possibilistic network classifiers. *Fuzzy Sets Syst.* 160(22), 3224–3238 (2009)
8. Higashi, M., Klir, G.J.: On the notion of distance representing information closeness. *International Journal of General Systems* 9(2), 103–115 (1983)
9. Jenhani, I., Ben Amor, N., Elouedi, Z., Benferhat, S., Mellouli, K.: Information affinity: A new similarity measure for possibilistic uncertain information. In: Mellouli, K. (ed.) *ECSQARU 2007*. LNCS (LNAI), vol. 4724, pp. 840–852. Springer, Heidelberg (2007)
10. Jenhani, I., Benferhat, S., Elouedi, Z.: Properties analysis of inconsistency-based possibilistic similarity measures. In: 12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based, Malaga, Spain, pp. 173–180 (June 2008)



11. Jenhani, I., Benferhat, S., Elouedi, Z.: On the use of clustering in possibilistic decision tree induction. In: Sossai, C., Chemello, G. (eds.) ECSQARU 2009. LNCS, vol. 5590, pp. 505–517. Springer, Heidelberg (2009)
12. Jenhani, I., Benferhat, S., Elouedi, Z.: Possibilistic similarity measures. In: Bouchon-Meunier, B., Magdalena, L., Ojeda-Aciego, M. (eds.) Foundations of Reasoning under Uncertainty. STUDFUZZ, vol. 249, pp. 99–123. Springer, Heidelberg (2010), [http://dx.doi.org/10.1007/978-3-642-10728-3\\_6](http://dx.doi.org/10.1007/978-3-642-10728-3_6)
13. Meng, L., van der Putten, P., Wang, H.: A comprehensive benchmark of the artificial immune recognition system (AIRS). In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS (LNAI), vol. 3584, pp. 575–582. Springer, Heidelberg (2005)
14. Murphy, M.P., Aha, D.W.: Uci repository databases (1996), <http://www.ics.uci.edu/mlearn>
15. Polat, K., Gunes, S.: Automated identification of diseases related to lymph system from lymphography data using artificial immune recognition system with fuzzy resource allocation mechanism (fuzzy-airs). Biomedical Signal Processing and Control 1(4), 253 (2006)
16. Polat, K., Gunes, S.: Principles component analysis, fuzzy weighting pre-processing and artificial immune recognition system based diagnostic system for diagnosis of lung cancer. Expert Systems with Applications 34(1), 214–221 (2008)
17. Polat, K., Sahan, S., Gunes, S.: Automatic detection of heart disease using an artificial immune recognition system (airs) with fuzzy resource allocation mechanism and k-nn (nearest neighbour) based weighting preprocessing. Expert Systems with Applications 32(2), 625–631 (2007)
18. Polat, K., Sahan, S., Kodaz, H., Günes, S.: A new classification method for breast cancer diagnosis: Feature selection artificial immune recognition system (FS-AIRS). In: Wang, L., Chen, K., S. Ong, Y. (eds.) ICNC 2005. LNCS, vol. 3611, pp. 830–838. Springer, Heidelberg (2005)
19. Sabri, F., Norwawi, N.M., Seman, K.: Hybrid of rough set theory and artificial immune recognition system as a solution to decrease false alarm rate in intrusion detection system. In: 2011 7th International Conference on Information Assurance and Security (IAS), pp. 134–138 (2011)
20. Sanguesa, R., Cabos, J., Cortes, U.: Possibilistic conditional independence: A similarity-based measure and its application to causal network learning. Int. J. Approx. Reasoning 18(1-2), 145–167 (1998)
21. Watkins, A.B.: Exploiting Immunological Metaphors in the Development of Serial, Parallel, and Distributed Learning Algorithms. Ph.D. thesis, University of Kent, Canterbury, UK (2005)
22. Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (airs): An immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines 5(3), 291–317 (2004), <http://dx.doi.org/10.1023/B%3AGENP.0000030197.83685.94>
23. Xu, L.: Biologically inspired intelligent fault diagnosis for power distribution systems. Ph.D. thesis, North Carolina State University (2006)
24. Zadeh, L.A.: Fuzzy sets as a basis for theory of possibility. Fuzzy Sets and Systems 1, 3–28 (1978)

# An Efficient Biomechanical Cell Model to Simulate Large Multi-cellular Tissue Morphogenesis: Application to Cell Sorting Simulation on GPU

Anne Jeannin-Girardon, Pascal Ballet, and Vincent Rodin

Univ. Bretagne Occidentale, UMR 6285, Lab-STICC, F-29200 Brest, France  
{anne.jeannin,pascal.ballet,vincent.rodin}@univ-brest.fr

**Abstract.** In the context of tissue morphogenesis study, *in silico* simulations can be seen as experiments in a virtual lab bench. Such simulations can facilitate the comprehension of a system, the test of hypotheses or the incremental refining of a model and its parameters. *In silico* simulations must be efficient and provide the possibility to simulate large tissues, containing thousands of cells. We propose to study tissue morphogenesis at the cellular level using our virtual biomechanical cell model. This model is based on a mass/spring system and coupled to a multi-agent system. We validated the relevance of our model through a case study: a cell sorting. Moreover, we took advantage of the large parallelism offered by graphics processing units (GPU), which contain up to thousands of cores: we implemented our model with the OpenCL framework. We ran large scale simulations, with up to  $10^6$  of our virtual cells. We studied the performance of our system on a CPU Intel Core i7 860, and two GPUs: a NVidia GeForce GT440 and a Nvidia GeForce GTX 690. The absence of synchronization in our implementation allowed the full benefits of the parallelism of these hardwares.

**Keywords:** virtual cell, multi-agent systems, virtual biology, tissue morphogenesis, OpenCL, high-performance simulation.

## 1 Introduction

The study of tissue morphogenesis requires the consideration of different temporal and spatial scales: spatial scales range from  $10^{-9}m$  (gene/protein) to  $10^0m$  (organism); temporal scales range from  $10^{-3}s$  (molecular interactions) to  $10^9s$  (lifetime). Numerical simulation cannot take all these scales into account, but two major approaches help their modeling: the top-down approach, in which individuals are considered as population, and the bottom-up approach, in which individuals are explicitly modeled and whose interactions give rise to emergent behaviors. Between these two approaches, cell centered models seem appropriate models to simulate cellular tissue [21]: between microscopic and macroscopic scales, the biological cell can on one hand modulate genes expression and, on the other hand, constitute tissues when considered as group.

Emergent approaches have the advantage of allowing modelers to be in the shoes of experimentators. The simulation thus becomes a virtual lab bench, making easier for modelers to refine their models and their parameters, to test some hypothesis: this is the exploratory aspect of emergent approaches. Besides, the cost of *in silico* experiments is much more attractive than this of *in vitro* or *in vivo* experiments. This is especially true with the use of architectures such as GPUs (*Graphics Processing Units*): not only are they powerful enough to simulate large scale multi-agent systems thanks to the large number of compute units they contain; but they also are very affordable compared to other parallel architectures like computer clusters.

The main contribution of this paper is to present an efficient biomechanical cell model to study tissue morphogenesis. We implemented this model on GPU with the OpenCL framework, allowing us to run large scale simulations. This parallel implementation was rather straightforward, thanks to the structure of our model.

The paper is organized as follows: section 2 gives an overview of existing cell models and their applications. Section 3 details our virtual cell model and its coupling to a multi-agent system while section 4 details the parallel implementation of our model. Section 5 gives some validation elements of our model through a case study: a cell sorting. A performance study shows the good performance of our implementation. Finally, section 6 draws conclusion about this work and gives some work perspectives.

## 2 Related Works

Although various cell models exist, they were not necessarily built to simulate large groups of cell, but rather cell internal dynamics: for instance, Virtual Cell [13] intends to simulate cell biological processes, such as membrane diffusion. The E-Cell software [20] simulates cells biochemical and genetic processes: it allows user to describe protein functions, protein-protein interactions, protein-DNA interactions, etc. ChemCell [14] is used to study spatial effects of proteins in the biological cell. However accurate, these models are too complex and are not made to simulate tissue.

Tissue simulation does not necessarily require a detailed description of intracellular processes. In particular, multi-agent systems are well suited to model cellular tissues and morphogenesis. The agents do not take internal dynamics into account, but rather describe cell behaviors (such as mitosis, cell differentiation, etc.) at a higher level. Cell2Organ [4], for example, is a cell-centered model used to generate artificial creatures from a single cell. Cells contain sensors in order to measure substrate concentration in the environment. These substrates are issued from an artificial chemistry integrated in the environment. The implementation is multi-threaded: each cell is ran by a thread and an additionnal thread manages the diffusion system. The authors of [3] propose a biomechanical cell model. The cells are built using two types of grains: the first one models the intracellular structure of cells: while in contact, they do not adhere with each other; the

second one models the membrane: those grains are linked together with elastic links. Cell growth is achieved by adding grains in the internal structure, meaning that the number of grains interacting in each cell increases as the cell grows, also increasing the computation time. The model has been partially parallelized using OpenMP. CellSys [9] is a software used for multi-cellular systems simulation. It implements a cell model that consists of an elastic sphere, which can migrate, divide and grow. These behaviors are modeled by equations that are solved using a multi-threaded equation solver. The authors of [5] propose a parallel platform for agent-based simulation. In this case, their cell model takes intra-cellular processes into account through a third party software and the simulations are executed on a computer cluster. FlameGPU [15] is a parallel version of the generalist framework Flame, used to simulate multi-agent systems. They simulated tissue growth based on this framework. However they focus on the underlying generalist structure of their agent rather than on an accurate model to study tissue morphogenesis. The Cellular Potts Model (CPM) [8] has been widely used in order to model and study morphogenesis processes. For instance, in [11], a CPM is used to study *Dictyostelium discoideum* self organization. CPM's dynamics is mainly based on the computation of surface, volume and contact energy. Finally, there are also a few models that focus on the study of mechanotransduction during morphogenesis. We can cite the work presented in [19], whose authors have studied *in silico* plant morphogenesis based on a mechanical model. The authors of [17] also studied plant morphogenesis through their model, in which cells can take mechanical signals into account. However, concerning the last two models, no insight was provided regarding the ability of these models to simulate important groups of cells (thousands of cells).

In our work, we focus on multi-agent systems (MAS), which allow fine grained description of the system and lead to complex emergent behaviors. Moreover, MAS are distributed systems and are thus adapted to a parallel implementation. The agents of our system are biomechanical cells which are modeled with a mass/spring system; we do not model intra-cellular processes in order to simplify the model, but rather describe cell behaviors at a higher level. We implemented our model on GPU hardware, considering that they are the most powerful kind of parallel platform available regarding their price and that they are handier than computer clusters.

### 3 Our Virtual Cell Model

Basic biological cell behaviors comprise: motility, adhesion, differentiation, molecules production/consumption, mitosis and apoptosis. In this article however, and given the case study we present, we focus only on cell motility and adhesion. Our cell model is based on the deformable cell model proposed in [1]. They successfully simulated cell migration on an adhesive substrate. We extended and improved this model in order to study large multi-cellular tissue morphogenesis thanks to the coupling of the cell model with a multi-agent system. The virtual cells are the agents of our system. The physics of the cells is described in the next subsection.

### 3.1 Cell Physics

Our virtual cell is a mass/spring system composed of seven nodes: one central node  $C$  and  $n = 6$  nodes  $N_i$ ,  $0 \leq i < n$ , on the membrane. These nodes are linked together into three main structures (Fig. 1a):

- the cytoskeleton (black links in Fig. 1a) is the set of links  $C \leftrightarrow N_i$  where  $C$  is the central node and  $N_i$ ,  $0 \leq i < n$ , are the membrane nodes;
- the cortex (orange links in Fig. 1a) is the set of links  $N_i \leftrightarrow N_{(i+2)\%n}$
- the membrane is the set of links  $N_i \leftrightarrow N_{(i+1)\%n}$  (green dashed links in Fig. 1a).

These structures are a simplified representation of biological cells that allow cell deformations such as cell compression or stretching, as illustrated in Fig. 1b. The virtual cell structure also helps preserving the physical integrity of the cell during simulations.

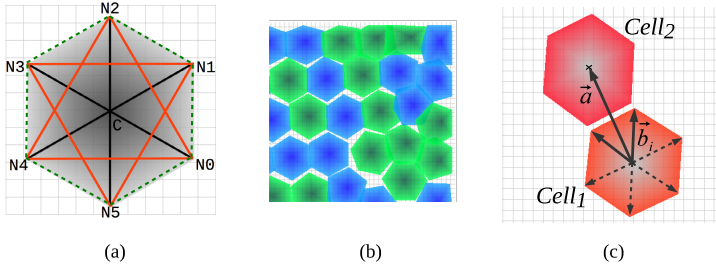
A restoring force is applied between the links of the cell's structure. Each link is a spring of stiffness  $k$  with an equilibrium length  $l_0$ . On each spring, the following restoring force is applied:  $\mathbf{F}_r = k(l - l_0)\mathbf{u} = k\Delta l\mathbf{u}$ . The unit vector  $\mathbf{u}$  denotes the direction of the force. The virtual cell also interacts with both the environment and its neighboring cells.

*Environmental Interactions.* So far, we modeled only one force issued by the environment: a Langevin force  $\mathbf{F}_l = I\mathbf{u}$  where  $I$  is the intensity of the force, applied on every node.

*Cell-cell Interactions.* Cell-cell interactions consist of two different forces: first a repulsive force is applied on a cell if an other cell is too close: if the distance  $l$  between the two cells' centers is less than a distance  $d_{min}$ , a spring is created between the two nodes, allowing the cells to repulse each other. The force caused by the spring is  $\mathbf{F}_{rep} = k(l - d_{min})\mathbf{u}$ .

Then an adhesive force  $\mathbf{F}_a = k(l - d_{max})\mathbf{u}$  allows the cells to adhere together given the adhesive coefficient between them. Here again, a spring allows the modeling of an adhesive force. If the distance between a selected membrane node and the center node is less than  $d_{max}$  ( $d_{max}$  is the sum of the adhesive coefficient and the cell radius), a restoring force is applied in order to bring the cells closer. If this distance is too large, the link breaks. If the distance is too small, a repulsion takes place. This way, we make sure that the cell's centers keep a minimal distance between them.

We select the nodes for adhesion in the following manner: springs are created between some membrane nodes of a first cell  $Cell_1$  and the center node of the second cell  $Cell_2$ : let  $C_1$  (resp.  $C_2$ ) be the center of  $Cell_1$  (resp.  $Cell_2$ ). Given that  $N_i^1$  ( $0 \leq i < n$ ) are the nodes of  $Cell_1$ , we define  $\mathbf{b}_i = \mathbf{C}_1 N_i^1$  and  $\mathbf{a} = \mathbf{C}_1 \mathbf{C}_2$ . The nodes of  $Cell_1$  selected for the adhesion are such that the dot product  $\mathbf{a} \cdot \mathbf{b}_i$  is greater than 0, as illustrated on Fig. 1c. Thanks to this method, that does not require either much computation time nor memory (as opposed to searching the adequate nodes and memorizing them), only  $Cell_1$ 's closest nodes to  $Cell_2$ 's center are taken into account to adhere to  $Cell_2$ .



**Fig. 1.** (a) The virtual cell is made of seven nodes which are linked together into three structures: the cytoskeleton (black links), the cortex (orange links), and the membrane (green dashed links). (b) Example of cell deformation: cells are compressed against each other and against the edge of the environment. (c) Selection of nodes for the adhesion between cells: if  $\mathbf{a} \cdot \mathbf{b}_i > 0$ , the node is selected for the adhesion

*Numerical Integration.* Once we computed all the forces applied on each node, these forces are integrated using the Euler method: we cope with the drawbacks of this method by 1) considering the environment as overdamped, in order to limit numerical instability; 2) choosing a time step that is small enough to stay in the domain of validity of the function to approximate. By limiting the drawbacks of this method, we take advantage of its light computation time, allowing more efficient simulations. Given the second Newton's law,  $\Sigma \mathbf{F} = \mathbf{F}_l + \mathbf{F}_r + \mathbf{F}_{rep} + \mathbf{F}_a = m\ddot{\mathbf{x}}$ , where  $m$  is the mass of one node (all nodes have the same mass), the integration allows the computation of the next position of a node  $n$  of a cell  $k$ :

$$\mathbf{x}_{k,n}(t+1) = \frac{\mathbf{F}_{k,n}}{m} \Delta t^2 + \dot{\mathbf{x}}_{k,n}(t) + \mathbf{x}_{k,n}(t) \quad (1)$$

Given that the environment is overdamped – we consider that  $\dot{\mathbf{x}}_{k,n}(t) = 0$  – the equation becomes:

$$\mathbf{x}_{k,n}(t+1) = \frac{\mathbf{F}_{k,n}}{m} \Delta t^2 + \mathbf{x}_{k,n}(t) \quad (2)$$

Table 1 presents the parameters of our system.

**Table 1.** Mass/spring system parameters [12]

Springs stiffness $k$	1 to 3 $\mu\text{N} \cdot \mu\text{m}^{-1}$
Cytoskeleton's springs equilibrium length $l_0$	5 $\mu\text{m}$
Node mass $m$	1 $\text{pg}$
Integration step $\Delta t$	0.3 $\text{s}$
$d_{min}$	10 $\mu\text{m}$
$d_{max}$	variable given the adhesive coefficients between cell types

The model as we built it allows an efficient parallel implementation: this implementation is presented in the next section, after a short introduction to the OpenCL framework.

## 4 Parallel Implementation of the Virtual Cells

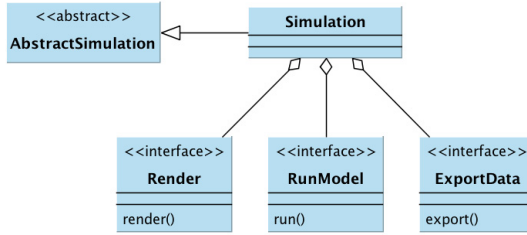
Multi-agent systems are well suited for parallel implementations given their distributed nature. We choose to take advantage of the computational power of GPU (*Graphics Processing Units*). GPUs can be considered as the most affordable parallel architecture today. Frameworks such as CUDA or OpenCL are dedicated to their programming. We use the framework OpenCL because unlike CUDA, it allows the programming of a large range of devices: CPU (Intel, AMD, ARM), GPU (both ATI's and NVIDIA's chips), IBM's Cell processor and even FPGAs. OpenCL programs are thus portable, and can be executed on various architectures.

*The OpenCL Framework.* OpenCL programs are made of 1) a host program, managing the parallel computation (mainly by choosing a compatible device, allocating memory and scheduling parallel code) and 2) the parallel code itself, whose functions are called kernels, and are programmed using a dedicated language, based on C. The OpenCL execution model is made of a grid organized in groups (called "work-groups") that contain instances of kernels (called "work-items"): one instance is a thread and in our case, each thread contains the behavior of one agent.

*Software Architecture for Parallel MAS Implementation.* Most of the host code is redundant from a model implementation to another. That is why we use a generalist software architecture we have conceived for parallel multi-agent systems simulation with OpenCL (Fig. 2) [10]. This architecture helps dealing with GPU and OpenCL constraints, such as memory accesses or the absence of dynamical memory allocation.

*Virtual Cell Implementation.* Each agent of the system has a unique identifier, and is represented by a kernel instance on the parallel device. Information about the agents are stored into structures of arrays. They are adapted data structure for GPU since they ensure coalescing access (consecutive threads access consecutive memory positions). Each agent accesses its own data only, it should not modify an other agent's data. Agent's data are the position of their nodes, the forces to be applied on each nodes and its type (in this article, the type is only a color), see Fig. 3. Agents are located in the environment, which is a two dimensional matrix: the identifier of the agent is stored in this matrix at the position of the central node. This allows an easy scan of the local environment of an agent in order to detect neighboring agents instead of having to build the list of neighbors with costly algorithms.

We built our model in such a way that so far, we did not use synchronization mechanisms such as atomic operations thanks to the mass/spring systems which



**Fig. 2.** UML graph of our software architecture. A simulation inherits general parameters from the `AbstractSimulation` class and instantiates modules specifically implemented for the simulation. The render module uses OpenGL primitives to display 2D or 3D graphics (OpenCL and OpenGL can share buffers on the device chosen for execution, meaning that no data are transferred between the rendering and the computation); the module “RunModel” is the OpenCL implementation of the model and the coordinating host code. Finally, the module “ExportData” allows data recording during the simulation in order to make qualitative analysis

```

struct data{
    ushort state[NB_AGENTS];
    ushort type[NB_AGENTS];
    ulong4 random[NB_AGENTS];
};

struct nodes{
    float4 centers[NB_AGENTS];
    float4 nodes0[NB_AGENTS];
    float4 nodes1[NB_AGENTS];
    float4 nodes2[NB_AGENTS];
    float4 nodes3[NB_AGENTS];
    float4 nodes4[NB_AGENTS];
    float4 nodes5[NB_AGENTS];
};
  
```

**Fig. 3.** Architecture’s C data structure adapted for our virtual cell model. Data types are built-in OpenCL data types. In particular, `typeN` are vector types, kind of equivalent to `type data[N]`. `NB_AGENTS` is the maximum number of agents to simulate. The structure `struct data` stores general information about agents. The fields `state` and `type` are default information; the field `random` is used as seeds for random number generation. The structure `nodes` stores the cells’ nodes position in the environment. This structure is also used to store the forces applied on each node

prevents two cells from being in the same environment location at the same time. Agent’s behaviors are implemented in OpenCL kernels, which are scheduled and launched by the host program: cell’s behavior consists of two kernels: one kernel computes the forces applied on the cell’s nodes; the second one integrates the forces with the Euler method (see section 3.1).

## 5 Case Study: Cell Sorting Due to Differential Adhesion

Cell sorting is a phenomenon occurring during tissue formation. One of the main hypothesis explaining cell sorting is the differential adhesion hypothesis (DAH), proposed in [18]. DAH states that cells rearrange themselves according to the adhesion between the different types of cell. For example, in the case of hydra tissue morphogenesis, such as studied in [16], an external layer of ectodermic



cells surrounds an aggregate of endodermic cells: it means that endodermic cells are more cohesive than ectodermic cells. Different works have studied the re-arrangement of cells from random aggregates. Most computational simulations were made with a Cellular Potts Model (CPM). In [8], an extended CPM is used to simulate cell sorting. The authors of [22] also used a CPM to investigate cell sorting parameters of their model, in particular the expression of cadherin, the molecule responsible for cell-cell adhesion, and the motility of the cells. Other authors investigate cell sorting based on chemotaxis, such as in [6]. However, simulations such as Graner and Glazier’s have shown that random motility coupled to DAH is sufficient for a random aggregate to sort. The authors of [2] proposed a self-propelled particle model to study cell motility and differential adhesion during cell sorting.

### 5.1 Implementation

The motility of cells is modeled by a Langevin force (taking into account both the movements of cells and their collisions with environmental molecules) whose intensity vary between 1 and  $10\mu N$ . Table 2 summarizes the adhesion between the two types of cell. Given that  $ad_{type1,type2}$  is the adhesion between two cells of type  $type1$  and  $type2$ , respectively, these values respect the following rule:

$$ad_{blue,blue} < ad_{blue,green} = ad_{green,blue} < ad_{green,green}.$$

In practice, these values are the minimal distance, in  $\mu m$  at which the cells adhere. The longer the distance, the stronger is the adhesion between two cells.

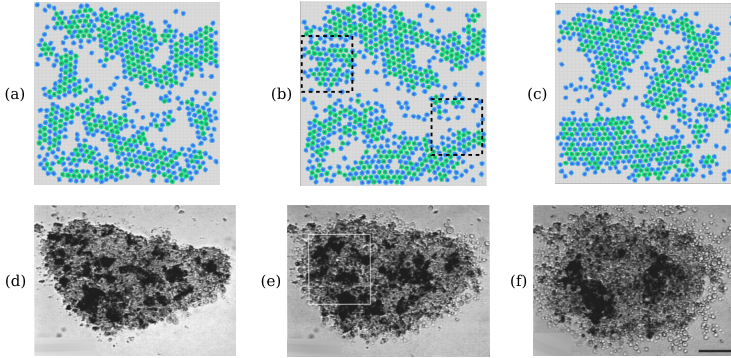
**Table 2.** Differential adhesions between the two types of cell

	Green	Blue
Green	2.5	2.1
Blue	2.1	1.4

Initially, the cells form a group in which types are randomly attributed, with approximatively the following proportions: green 40% and blue 60%. What is expected during a simulation is that green cells regroup together and that blue cells surround the green aggregates. Some results are discussed below.

### 5.2 Results and Discussion

Figure 4 is an example of cell sorting simulation. Figures from (a) to (c) are extracted from one of our simulations, in which 700 cells where simulated. In Fig. 4b, we show the integration of two small clusters (see Fig. 4a) to a larger one. In Fig. 4c, we see large solid clusters, but also, on the right, that small groups of cells can escape a cluster: this shows the difficulties in setting the parameters of the MAS (the intensity of the Langevin force and the adhesion). We also show hydra’s cellular aggregates, taken from [16], to illustrate the likeness of



**Fig. 4.** Example of a cell sorting simulation. Images of the first row are extracted from the simulation of 700 cells. From (a) to (c): along the simulation, clusters are formed thanks to differential adhesion and cell motility. Images of the second row are real cellular aggregates taken from [16]. From (d) to (f), one observes cluster formation: endodermal cells (dark cells) aggregates and are slowly surrounded by ectodermal cells (light cells)

our simulation with a real system. Our green cells are the equivalent of the endodermal cells. As the simulation runs, these cells form solid clusters. Blue cells are the equivalent of the ectodermal cells.

During simulation, we observe the sorting of cells thanks to their motility and the DAH. However, there is some comments to make about the model as it is so far, mainly about the tissue consistency: instead of having a whole tight group of cells, we can see that cells tend to form separate groups. This is due to the delicate balance between the intensity of the Langevin force and the adhesive rate of cells. Adding some adhesitivity for the blue cell will cause cell groups to be too solid to be “broken” in order to rearrange the group. Automatic parameters exploration may be useful to tune the parameters of the MAS, and quantifying the degree of cell sorting would help the design of a fitness function of a genetic algorithm in order to converge towards cell sorting.

### 5.3 Performance

In order to study the scalability of our system, we ran some sets of experiments. We used three devices: an CPU (Intel Core i7 860) and two GPU: a NVidia GeForce GT 440, which is a low-end GPU and a NVidia GeForce GTX 690 which is a high-end GPU. The GTX 690 is actually a bi-GPU; however we use only one of the two devices in our experiments. The main characteristics of these three devices are summarized in Table 3.

*Results.* We measured the average execution time of the OpenCL code over 1000 simulation steps. For that, we use OpenCL events coupled with the function `clGetEventProfilingInfo` in order to measure the kernels execution time. There were no memory transfer between the host and the device during the

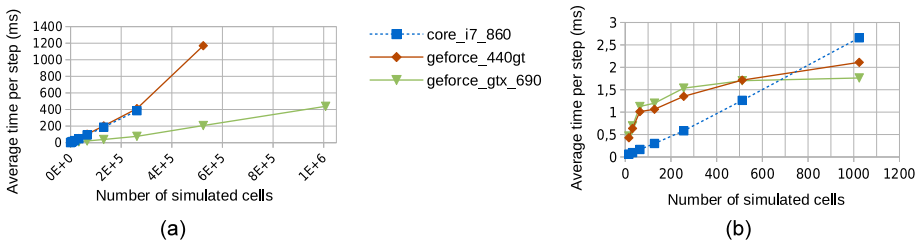
**Table 3.** Main characteristics of the devices used to measure the performance of the system as provided by the devices' respective datasheets

	Intel Core i7 860	NVidia GT 440	NVidia GTX 690
Frequency	2.80 GHz	1600 MHz	900 - 1000 MHz
Processing elements	8 (4 compute units)	96 (2 compute units)	1536 per GPU (8 compute units)
Shared memory size configurations	32KB	up to 48KB	up to 48KB

computation (except before the execution started, in order to load the data on the device). We ran several simulations, varying the number of simulated cells: Fig. 5a illustrates the global performance of our system on the three devices: as expected, the GTX 690 shows good performance: it does 5.66 times better than the GT 440 to simulate  $5.10^5$  cells. Up to  $2.10^5$  cells, the CPU performs similarly as the GT 440. The GTX 690 is the only device that could simulate over one million cells: the CPU could not execute more than  $2.10^5$  cells and the GT 440 could not execute more than  $5.10^5$  cells: our model has an important memory footprint. The hardware, along with its OpenCL implementation may prevent us from allocating as much memory as we need for very large systems (here, we are talking about around  $5.10^5$  cells). However, we cope with that limit given that for now, we are more focused on refining our model than on increasing the number of cells we can simulate.

Figure 5b shows the same data than Fig. 5a, but it only shows the execution of 1000 cells and less. It shows the good performance of the CPU for a number of cells that does not exceed 700: this is due to the conditional branches in the OpenCL code. However, when the number of simulated cells increases, the parallelism of the device becomes more important than its ability to execute conditional code.

These examples show that our system has a large scalability: we simulated up to  $10^6$  cell, and the use of a high performance GPU allows very interesting computation time, although in practice the overall computation time also depends



**Fig. 5.** (a) Average execution times of a simulation step for the Intel Core i7 860 (blue curve), the NVidia GT 440 (red curve) and the NVidia GTX 690 (green curve). (b) Average execution time of a simulation step for less than 1000 simulated cells, showing the excellent performance of the Core i7 compared to the two GPUs

1) on the host device, which has to be able to perform a quick scheduling of the kernels or may have some data to process between two kernel calls and 2) on the fact that we do or we do not transfer data from and back the device to the host during the computation. Generally speaking, SIMD architecture such as GPU are well suited to run reactive agent-based simulations, given the distributed nature of these systems. So far, our cell model does not require synchronization, allowing the full benefits of the parallelism and resulting in good performance in execution. The CPU Intel Core i7 860 performs better than the two GPUs for less than 700 simulated cell. The NVidia GeForce GT 440 performs quite poorly compared to the NVidia GeForce GTX 690, which could simulate  $10^6$  cells. The use of OpenCL allowed us to execute simulations on various devices.

## 6 Conclusion

In this paper, we have presented our virtual biomechanical cell model designed to investigate *in silico* tissue morphogenesis: this model is based on a mass/spring system and coupled to a multi-agent system. Simulating tissues requires important computing resources, this is why we implemented our model in a MAS dedicated software architecture that uses OpenCL. This implementation allows us to take advantage of parallel hardware. We studied the performance of our system on three devices: an Intel Core i7 860, a NVidia GeForce GT 440 and a NVidia GeForce GTX 690. The model does not require synchronization mechanism, allowing the full benefits of parallelism: we simulated tissues containing up to  $10^6$  cells.

As a first step, we tested the relevance of our system using a cell sorting simulation. Although the result could have been more convincing (especially compared to Cellular Potts Model's cell sorting simulations), we observed that cell aggregates were formed thanks to cell-cell adhesion and random cell motility. Fine parameters tuning is difficult when it comes to multi-agent system, and a perspective of work would be to design an algorithm that would allow automatic parameters exploration of the system. Nevertheless, our major perspective is to complete our model by 1) defining other fundamental cell behaviors: mitosis, differentiation, molecules consumption and production, and 2) making the cells respond to mechanical stress such as compression, stretching and shearing in order to study the influence of mechanotransduction during tissue morphogenesis [7].

**Acknowledgements.** This work was supported by the Région Bretagne (France).

## References

1. Ballet, P., Tracqui, P.: Deformable virtual cell migration: biomechanical and multi-agent modeling of cell migration-agent modeling of cell migration. RSTI, TSI Series (Special issue "Modeling and simulation for post-genomics") 26(1-2/2007) (2007)
2. Belmonte, J.M., Thomas, G.L., Brunnet, L.G., de Almeida, R.M.C., Chaté, H.: Self-propelled particle model for cell-sorting phenomena. *Physical Review Letters* 100, 248702 (2008)
3. Chélin, Y., Azzag, K., Canadas, P., Averseng, J., Baghdiguian, S., Maurin, B.: Simulation of cellular packing in non-proliferative epithelia. *Journal of Biomechanics* 46(6), 1075–1080 (2013)

4. 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 (2008)
5. Da-Jun, T., Tang, F., Lee, T., Sarda, D., Krishnan, A., Goryachev, A.: Parallel computing platform for the agent-based modeling of multicellular biological systems. In: Liew, K.-M., Shen, H., See, S., Cai, W. (eds.) *PDCAT 2004*. LNCS, vol. 3320, pp. 5–8. Springer, Heidelberg (2004)
6. Eyiurekli, M., Lelkes, P.I., Breen, D.E.: A computational system for investigating chemotaxis-based cell aggregation. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) *ECAL 2007*. LNCS (LNAI), vol. 4648, pp. 1034–1049. Springer, Heidelberg (2007)
7. Fleury, V.: A change in boundary conditions induces a discontinuity of tissue flow in chicken embryos and the formation of the cephalic fold. *The European Physical Journal E* 34(7), 1–13 (2011)
8. Graner, F., Glazier, J.A.: Simulation of biological cell sorting using a two-dimensional extended potts model. *Physical Review Letters* 69(13), 2013–2016 (1992)
9. Hoehme, S., Drasdo, D.: A cell-based simulation software for multi-cellular systems. *Bioinformatics* 26(20), 2641–2642 (2010)
10. Jeannin-Girardon, A., Ballet, P., Rodin, V.: A software architecture for multicellular system simulations on graphics processing units. *Acta Biotheoretica*, 1–11 (2013), <http://dx.doi.org/10.1007/s10441-013-9187-3>
11. Marée, A.F.M., Hogeweg, P.: How amoeboids self-organize into a fruiting body: Multicellular coordination in dictyostelium discoideum. *Proceedings of the National Academy of Sciences* 98(7), 3879–3883 (2001)
12. Mogilner, A., Oster, G.: Force generation by actin polymerization ii: The elastic ratchet and tethered filaments. *Biophysical Journal* 84(3), 1591–1605 (2003)
13. Moraru, I.I., Schaff, J.C., Slepchenko, B.M., Blinov, M.L., Morgan, F., Lakshminarayana, A., Gao, F., Li, Y., Loew, L.M.: Virtual Cell modelling and simulation software environment. *IET Systems Biology* 2(5), 352–362 (2008)
14. Plimpton, S.J., Slepoy, A.: Microbial cell modeling via reacting diffusive particles. *Journal Of Physics* 16, 305–309 (2005)
15. Richmond, P., Walker, D., Coakley, S., Romano, D.: High performance cellular level agent-based simulation with flame for the gpu. *Briefings in Bioinformatics* 11(3), 334–347 (2010)
16. Rieu, J.P., Kataoka, N., Sawada, Y.: Quantitative analysis of cell motion during sorting in 2d aggregates of hydra cells. *Physical Review E* 57, 924–931 (1998)
17. Rudge, T., Haseloff, J.: A Computational Model of Cellular Morphogenesis in Plants. In: Capcarrère, M.S., Freitas, A.A., Bentley, P.J., Johnson, C.G., Timmis, J. (eds.) *ECAL 2005*. LNCS (LNAI), vol. 3630, pp. 78–87. Springer, Heidelberg (2005)
18. Steinberg, M.: Reconstruction of tissues by dissociated cells. *Science* 141(3579) (1963)
19. Stoma, S., Chopard, J., Godin, C., Traas, J.: Using mechanics in the modelling of meristem morphogenesis. In: *5th International Workshop on Functional-Structural Plant Models*, Napier, New Zealand, pp. 52, 1–4 (2007)
20. Tomita, M., Hashimoto, K., Takahashi, K., Shimizu, T.S., Matsuzaki, Y., Miyoshi, F., Saito, K., Tanida, S., Yugi, K., Venter, J.C., III, C.A.H.: E-cell: software environment for whole-cell simulation. *Bioinformatics* 15(1), 72–84 (1999)
21. Walker, D.C., Southgate, J.: The virtual cell - a candidate coordinator for middle-out modelling of biological systems. *Briefings in Bioinformatics* 10(4), 450–461 (2008)
22. Zhang, Y., Thomas, G.L., Swat, M., Shirinifard, A., Glazier, J.A.: Computer simulations of cell sorting due to differential adhesion. *PLoS ONE* 6(10) (2011)

# Computational Modelling of the Interruptional Activities between Transposable Elements

Lingling Jin and Ian McQuillan\*

Department of Computer Science, University of Saskatchewan,  
110 Science Place, Saskatoon, SK, Canada  
lingling.jin@usask.ca, mcquillan@cs.usask.ca

**Abstract.** Transposable elements (TEs) are DNA sequences that can either move or copy themselves to new positions within a genome. They constitute approximately 45% of the human genome. Knowing the evolution of TEs is helpful in understanding the activities of these elements and their impacts on genomes. In this paper, we devise a formal model providing notations/definitions that are compatible with biological nomenclature, while still providing a suitable formal foundation for computational analysis. We define sequential interruptions between TEs that occur in a genomic sequence to estimate how often TEs interrupt other TEs, useful in predicting their ages. We also define the recursive interruption context-free grammar to capture the recursive nature in which TEs nest themselves into other TEs. We then associate probabilities to convert the context-free grammar into a stochastic context-free grammar, and discuss how to use the CYK algorithm to find a most likely parse tree predicting TE nesting.

**Keywords:** transposable elements, stochastic context-free grammars, interruptional analysis, formal modelling.

## 1 Introduction

In humans, coding sequences comprise less than 5% of the genome, whereas repeat sequences account for at least 50% and probably much more [11], the majority of which are *transposable elements* (TEs), or *transposons*, first discovered by McClintock in 1949 in *Zea mays*. TEs are interspersed DNA sequences that can move or transpose themselves to new positions within the genome. They are found in nearly all species (both prokaryotes and eukaryotes, such as bacteria, fungi, plants and animals) that have been studied and constitute a large fraction of some genomes [5]. The human genome is particularly rich in TEs, at about 45% of the genome [12]. Human transposable elements have been reported to cause human diseases, including several types of cancer, such as breast cancer, colon cancer, retinoblastoma, neurofibromatosis, hepatoma, etc., through insertional mutagenesis of genes critical for preventing or driving

---

\* This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

malignant transformation [2]. Non-LTR retrotransposons are the predominant source of TE-related mutagenesis in the human genome. Thus, it is meaningful to understand the activities and evolution of TEs.

According to their mechanism of transposition, TEs are traditionally classified into two major classes: Class I elements are those that transpose via an RNA intermediate using a “copy-and-paste” mechanism, referred to as “*retrotransposons*”; Class II elements, which are DNA-mediated using a “cut-and-paste” mechanism, are often called “*DNA transposons*”.

TEs are also described as being *autonomous* or *non-autonomous* based on whether or not they encode their own genes for transposition. Those transposons that possess a complete set of transposition protein domains are called *autonomous*. However, the term autonomous does not imply that an element is active or functional. Transposons that clearly lack an intact set of mobility-associated genes are called *non-autonomous*, whose transposition requires participation of one or more proteins encoded by an autonomous element. Within each of these classes, TEs can be further subdivided into several types on the basis of the structural features of their sequences. In general, the information of each type of TE is shown in Table 1 (information from [11]).

**Table 1.** The information of each type of transposable elements in the human genome

TE Type	TE Class	Mode of Transposition	Length	Copy Number	Frac. of Genome
LINES	Retrotransposons	Autonomous	6-8 kb	850,000	21%
SINEs	Retrotransposons	Non-autonomous	100-300 bp	1,500,000	13%
LTR retrotransposons	Retrotransposons	Autonomous	6-11 kb	450,000	8%
		Non-autonomous	1.5-3 kb		
DNA transposons	DNA transposons	Autonomous	2-3 kb	300,000	3%
		Non-autonomous	80-3,000 bp		

Each TE has had a distinct period of transpositional activity when it is active, in which it has spread through the genome, followed by inactivation and accumulation of mutations (they can also mutate while active). The result of the transpositional activities, over eons, can be described by (summarized from [4]): (1) older TEs are heavily interrupted by younger TEs, but have not inserted into younger elements; (2) younger TEs, with a relatively recent period of activity, have inserted into older elements that were present in the genome, but are not interrupted by older elements; (3) elements of intermediate age have both inserted into older elements and been themselves fragmented by younger elements. In addition, younger TEs interrupt not only older TEs, but also the fragmented TEs that had been previously interrupted, which makes the interruptions in the sequence even more complex. That is, interruptions are nested together recursively [10].

A novel method was introduced in [4] to estimate TE ages in mammalian genomes based on the frequency with which TEs have inserted themselves into

other TEs. The resultant ordering obtained from a positional distribution agreed reasonably with published chronologies. This is in contrast to the more common divergence-based methods [9,1,13] to estimate TE ages.

In this paper, we will create a formal model capturing TE types and fragments within genomic sequences. Our model consists of initial definitions of TEs, the set of TEs, and the set of TE fragments. The model does not attempt to simulate or capture the molecular operations of TE movement (copying/cutting and pasting throughout a genome), which would create clear non-context-free patterns, making algorithmic analysis difficult. Rather, the model only describes the order and distance between TE fragments in genomic sequences by grouping homologous TEs together. At this level of abstraction, the model can be used to capture and calculate interruptions and their frequencies in a general way. In addition, we provide a stochastic context-free grammar to capture recursive TE nesting, allowing standard polynomial time parsing to be used to calculate a prediction of the nesting.

We attempt to make the model formal yet realistic and compatible with the biological literature on TEs. For this reason, the definitions are quite lengthy and make up a significant portion of this paper. However, we feel that it is necessary to contribute a suitable foundation for future computational analysis.

## 2 Formal Model of TE Fragments and Pruned Sequences

The purpose of this section is to develop a formal model of TEs and fragments of TEs in order to describe the biological concepts and problems clearly. We assume knowledge with context-free grammars [6] and parse trees, as well as common bioinformatics algorithms [7], such as pairwise and multiple alignments, and consensus sequences.

As a large part of research on bioinformatics is based on the analysis of DNA or amino-acid sequences, we will first briefly define a general sequence/string and other mathematical preliminaries in Definition 1.

**Definition 1.** *We define several terms and notations: An alphabet  $\Sigma$  is an abstract and finite set of symbols. A string is any finite sequence of characters over an alphabet. The length of a string  $s$ , denoted by  $|s|$ , is the number of characters in the string. The empty word is denoted by  $\lambda$  and is of length 0. The set of all strings (including the empty word) over  $\Sigma$  is denoted by  $\Sigma^*$ . Let  $\Sigma$  be an alphabet and  $s = s_1s_2\dots s_n$  be a string,  $s_i \in \Sigma$ ,  $1 \leq i \leq n$ , and  $j, k$  satisfy  $1 \leq j \leq k \leq n$ , then  $s(j, k) = s_js_{j+1}\dots s_k$ . Moreover,  $s(j) = s_j$ , is the  $j^{\text{th}}$  character alone. Let  $s \in \Sigma^*$ , then  $\text{frag}(s)$  is the set of all possible fragments (substrings) of  $s$ . That is  $\text{frag}(s) = \{x(p, q) \mid 1 \leq p \leq q \leq |s| \cup \{\epsilon\}\}$ . We extend this to sets of strings  $S \subseteq \Sigma^*$  by  $\text{frag}(S) = \bigcup_{s \in S} \text{frag}(s)$ . Given a set  $X$ , then  $|X|$  is the number of elements in  $X$ .*

When talking about a transposable element, life scientists usually are referring to a set of similar sequences that evolved from a single TE sequence. Therefore, we define a *transposable element* to itself be a set of strings (usually these strings



will be similar to each other). We also define a *set of TEs*, an *instance of a TE* and the *consensus TE* in Definition 2.

**Definition 2.** A transposable element (TE)  $X$  is a finite set of strings with  $X \subseteq \Sigma^*$ . An instance of a TE  $X$  is an element  $x \in X$ . A consensus TE is a consensus sequence of the elements of  $X$ . A set of TEs  $\chi$  is a finite set of TEs. That is,  $\chi \subseteq 2^{\Sigma^*}$ ,  $\chi$  is finite and each element of  $\chi$  is finite.

Most representative eukaryotic repetitive sequences have been compiled and reconstructed in a database called Repbase Update [8], which is a comprehensive database of the consensus sequences of repetitive elements (not only TEs, but also other repeats), that are present in diverse eukaryotic organisms.

Because of different biological contexts, it is also possible to interpret the set of TEs,  $\chi$ , in multiple ways depending on the purpose. For example, we could use as  $\chi$  the set of all TEs and TE instances that are present in a single genome, or as the set of sequences collected in Repbase Update, or any set of sequences that are all similar to the consensus sequences in Repbase Update within a threshold.

Knowing that one transposable element contains a number of instances, and each instance itself is a string, now we will define a *TE fragments set*. We expect to see many such fragments scattered throughout genomes as TEs become fragmented within a genome as they become interrupted by other TEs.

**Definition 3.** Let  $\chi$  be a finite set of TEs. Then we call  $\bar{\chi} \subseteq 2^{\Sigma^*}$  a TE fragments set if each element  $\bar{X} \in \bar{\chi}$  is a subset of  $\text{frag}(X)$ , for some  $X \in \chi$ .

Thus, after picking a set of TEs, a TE fragments set is any set where each element consists of only fragments of one transposable element. Then in principle, we can pick any number of fragment sets for one set of TEs.

Although we defined TE fragments sets in a general way, we would also like to create a restriction to transposable elements that occur in present-day sequences. RepeatMasker [14] is the predominant library-based tool used in repeat identification, which has become a standard tool for any search of repeats in genomes. It is a sophisticated program that uses precompiled repeat libraries to find copies of known repeats represented in the libraries. The program performs a similarity search on both the “+” and “-” DNA strands based on local alignments, then outputs masked genomic DNA and provides a tabular summary of repeat content detected in both DNA strands. We now can restrict Definition 3 with the aid of RepeatMasker as follows:

**Definition 4.** Let  $s$  be a string representing some genomic sequence and  $\chi_s$  be the set of TEs existing in  $s$ , then  $\bar{\chi}(s \xrightarrow{RM} \chi_s)$  is a RepeatMasker TE fragments set, running the program with a set of consensus TEs,  $\chi_s$ , against the genomic sequence  $s$ .

In other words, each element of  $\bar{\chi}(s \xrightarrow{RM} \chi_s)$  is a subset of some element of  $\bar{\chi}$ , where only TE fragments detected by the RepeatMasker program are selected.

For each TE fragment  $z$  in some  $\bar{X} \in \bar{\chi}(s \xrightarrow{RM} \chi_s)$ , we associate a tuple in Definition 5, whose attributes are referred to in our model, which is also consistent with the output of the RepeatMasker program.

**Definition 5.** Given a genomic sequence  $s$  and a set of TEs  $\chi_s$ , each TE fragment  $z$  in each  $\bar{X} \in \bar{\chi}(s \xleftrightarrow{RM} \chi_s)$  is a tuple:

$$\text{info}(z) = (\text{genoName}, \text{genoStart}, \text{genoEnd}, \text{genoLeft}, \text{strand}, \text{TEName}, \text{TEClass}, \text{TEStart}, \text{TEEnd}, \text{TELeft}). \quad (1)$$

We use the operator “.” to access the attributes, for example,  $z.\text{TEName}$  is the name of the TE to which fragment  $z$  belongs. The definition of each attribute is summarized in the list as follows (from [14]), and described in Example 1.

**genoName:** The name of genomic sequence.

**genoStart:** Start in genomic sequence.

**genoEnd:** End in genomic sequence.

**genoLeft:** Opposite number of bases after the matched alignment in genomic sequence.

**strand:** Relative orientation “+” or “-”.

**TEName:** Name of the TE.

**TEClass:** Class of the TE.

**TEStart:** Start in TE sequence, if strand is “+”; or opposite number of bases after the matched alignment in TE sequence, if strand is “-”.

**TEEnd:** End in TE sequence.

**TELeft:** Opposite number of bases after the matched alignment in TE sequence, if strand is “+”; or start in TE sequence, if strand is “-”.

Then, two TE fragments in the same set  $\bar{X} \in \bar{\chi}(s \xleftrightarrow{RM} \chi_s)$  have the same name. Also, a TE fragment can be present in either “+” or “-” strand, however, in both cases, we use the “+” strand coordinate to represent the location where it occurs. The orientations of the TE fragment are distinguished by the *TEStart* or *TELeft* attributes. For example, if a fragment  $z$  is in the “+” strand, then  $z.\text{TEStart} \geq 0$  and  $z.\text{TELeft} \leq 0$ , otherwise,  $z.\text{TEStart} \leq 0$  and  $z.\text{TELeft} \geq 0$ . In general, no matter in which strand a TE fragment occurs, our notations in the formal model are consistent. Example 1 picks two TE fragments showing the meanings of their attributes visually with respect to a genomic sequence and TE consensus sequences.

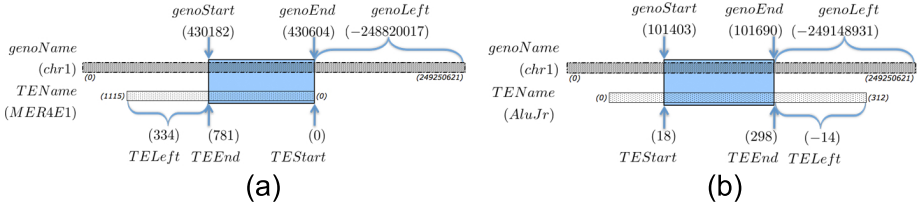
**Example 1.** Given two TE fragments taken from the output of RepeatMasker comparing the Human Genome<sup>1</sup> against the library of human transposable elements in Repbase Update, as listed in Table 2 with their detailed attributes.

**Table 2.** An example of two TE fragments in human chromosome 1

<i>genoName</i>	<i>genoStart</i>	<i>genoEnd</i>	<i>genoLeft</i>	<i>strand</i>	<i>TEName</i>	<i>TEClass</i>	<i>TEStart</i>	<i>TEEnd</i>	<i>TELeft</i>
chr1	430182	430604	-248820017	-	MER4E1	LTR	0	781	334
chr1	101403	101690	-249148931	+	AluJr	SINE	18	298	-14

Fig. 1 (a) illustrates a fragment of the transposon MER4E1, which was detected in the “-” strand of chromosome 1, and Fig. 1 (b) shows a fragment of the

<sup>1</sup> hg19, the Feb. 2009 assembly of the human genome.



**Fig. 1.** The conceptual visualization of the two TE fragments (in blue shadow) in Table 2. Note that the lengths of the visualized sequences in the figure are not proportional to their actual lengths.

*transposon AluJr, which was detected in the “+” strand of chromosome 1. Since the two fragments are detected in different strands, they are oriented oppositely.*

Most of the present-day copies of transposable elements are detected by locally aligning the consensus TE sequences against a DNA sequence, thus the DNA sequence is fragmented into segments by the matched local alignments. Some segments are detected as fragments of those TEs, while some are non-transposon DNA sequence. We then prune this sequence to present only the TE segments. This process is defined in Definition 6.

**Definition 6.** Let  $s$  be a genomic sequence,  $\chi_s = \{X_1, \dots, X_m\}$  a fixed ordering of the set of TEs in  $s$ , and  $\bar{\chi}_s$  a set of TE fragments. Assume  $s = w_0 z_1 w_1 z_2 \dots z_k w_k$ , with  $z_1, \dots, z_k$  in sets in  $\bar{\chi}_s$ , and no fragment of  $w_0, \dots, w_k$  in sets in  $\bar{\chi}_s$ . Then a pruned sequence  $\bar{s}$  of  $s$  with respect to  $\bar{\chi}_s$  is

$$\bar{s} = \beta_0 z_1 \beta_1 z_2 \beta_2 \dots z_k \beta_k, \text{ where } \beta_i = |w_i|, 0 \leq i \leq k. \quad (2)$$

That is, in a pruned sequence, we replace all non-TE fragments with their length.

In addition, from  $\bar{s}$  and  $\chi_s$ , we define an order pruned sequence  $\bar{s}_o$  of  $\bar{s}$  as the string over  $\{1, \dots, m\}^*$ ,

$$\bar{s}_o = j_1 j_2 \dots j_k, \text{ where } z_i \in X_{j_i}, \text{ for all } i, 1 \leq i \leq k. \quad (3)$$

We can also extend a pruned sequence to a set of pruned sequences. Let  $S = \{s_1, \dots, s_N\}$ , then the set of pruned sequences of  $S$  is  $\bar{S} = \{\bar{s}_1, \dots, \bar{s}_N\}$ .

So far, we have defined some fundamental concepts associated with key biological terms, such as a transposable element, a TE fragment, and a pruned sequence, and also extended them to sets. In the next section, we will move the emphasis to the dynamic interruptional activities between different TEs.

### 3 Sequential Interruptions

As discussed in Section 1, newer TEs tend to interrupt older TEs, thereby fragmenting older TEs within the single linear sequence. By analyzing that sequence,

we are able to predict where the insertional and transpositional activities occurred with the frequencies of the interruptions, and also predict an order that those activities occurred, and further infer the ages of these TEs. This is essentially the same approach created in [4] when calculating the *interruptional matrix* that counts the number of interruptions between each pair of TEs. They then predict relative ages of TEs by permuting the rows and columns of the matrix that achieves a lowest sum of upper triangles of the matrix, corresponding to reordering, from those that get interrupted most while interrupting least, to those that interrupt most while getting interrupted least. We can similarly capture this with our model as well. We will describe this matrix, as well as discuss other applications of our model throughout this section and Section 4.

To analyze interruptional patterns, we are only interested in TE fragments and their relative positions in a genomic sequence. In [4], they classify an interruption as occurring when one TE fragment is within a certain distance from a fragment on the left and a fragment on the right, where both are from the same TE, and the two fragments are “close to” continuous within the TE. This can be calculated from pruned sequences in Definition 6, and can provide all that is necessary to calculate our interruptional matrix. Before defining a sequential interruption in Definition 8, we will define *continuous TE fragments*.

**Definition 7.** Let  $s$  be a genomic sequence with a set of TEs  $\chi_s$ , TE fragment set  $\bar{\chi}(s \xleftrightarrow{RM} \chi_s)$  and pruned sequence  $\bar{s} = \beta_0 z_1 \beta_1 z_2 \dots z_k \beta_k$  as in Equation (2).

Then two TE fragments  $z_i$  and  $z_j$  ( $i < j$ ) are continuous TE fragments,  $z_i \stackrel{\varepsilon, E}{\sim} z_j$ , with threshold  $\varepsilon \in \mathbb{N}$  and distance  $E \in \mathbb{N}$ , if they satisfy the following conditions:

1. they belong to the same transposable element:  $z_i.TEName = z_j.TEName$ ;
2. they are detected in the same DNA strand:  $z_i.strand = z_j.strand$ ;
3. they are either separated or overlap<sup>2</sup> with less than or equal to a threshold,  $\varepsilon$ , with respect to the TE consensus sequence to which they belong:
 
$$\begin{cases} abs(z_j.TEStart - z_i.TEEnd) \leq \varepsilon, & \text{if } z_i \text{ and } z_j \text{ occur in the “+” strand.} \\ abs(z_i.TEStart - z_j.TEEnd) \leq \varepsilon, & \text{if } z_i \text{ and } z_j \text{ occur in the “-” strand.} \end{cases}$$
4. They are in the genomic sequence within a distance,  $E$ , of non-transposon DNA sequence:  $\sum_{r=i}^{j-1} \beta_r \leq E$ .

Notice that continuous TE fragments are not necessarily beside each other in the genomic sequence, as there can be a distance of  $E$  between them. Some continuous TE fragments appear to have a duplication of a portion of the transposon. This is because RepeatMasker often extends the homology match of both fragments to the TE consensus sequence by several base pairs.

**Definition 8.** Given a genomic sequence  $s$ , a set of TEs with a fixed ordering on its elements  $\chi_s = \{X_1, X_2, \dots, X_m\}$ , a threshold  $\varepsilon \in \mathbb{N}$  in TE consensus sequence, a distance  $E \in \mathbb{N}$  in genomic sequence, and a pruned sequence  $\bar{s} =$

---

<sup>2</sup> We calculate the amount that separate them or the amount they overlap using the  $abs()$  function to get the absolute value.

$\beta_0 z_1 \beta_1 z_2 \dots z_k \beta_k$ , as in as in Equation (2). We define sequential interruptions of  $X_j$  by  $X_i$  as

$$\Xi_s^{\epsilon, E}(X_i, X_j) = \{k \mid z_k \in \bar{X}_i, z_{k-\eta_1}, z_{k+\eta_2} \in \bar{X}_j, z_{k-\eta_1} \stackrel{\epsilon, E}{\sim} z_{k+\eta_2}, \text{ and } \eta_1, \eta_2 \in \mathbb{N}\}. \tag{4}$$

Thus  $X_i$  is called interrupter, and  $X_j$  is called interruptee.

The frequencies with which the interruptions between different TEs occur in the sequence can also infer the activities of these TEs. Therefore, we also define the abundance of interruptions to capture the frequencies of interruptions.

**Definition 9.** Given a genomic sequence  $s$ , a set of TEs with a fixed ordering on its elements  $\chi_s = \{X_1, X_2, \dots, X_m\}$ , the abundance that  $X_i$  interrupts  $X_j$  in  $s$ ,  $1 \leq i \leq m$ ,  $1 \leq j \leq m$ , is defined as the total number of times that  $X_i$  interrupts  $X_j$ . The abundance is equal to  $|\Xi_s^{\epsilon, E}(X_i, X_j)|$ .

For the genome  $S$  that has chromosomes  $s_1, s_2, \dots, s_N$ , we then add up the abundance that  $X_i$  interrupts  $X_j$  for all chromosomes, which is  $|\Xi_S^{\epsilon, E}(X_i, X_j)| = \sum_{n=1}^N |\Xi_{s_n}^{\epsilon, E}(X_i, X_j)|$ . The interruption array of  $X_i$  on  $S$ , for  $1 \leq i \leq m$ , is the array  $M(i) = [|\Xi_S^{\epsilon, E}(X_i, X_j)|]_{j=1, \dots, m}$ . The interruption matrix on  $S$  is an  $m \times m$  matrix defined by  $M = [|\Xi_S^{\epsilon, E}(X_i, X_j)|]_{\substack{i=1, \dots, m \\ j=1, \dots, m}}$ .

The interruption array and matrix are different ways to structure the abundance by using the ordering on the elements in  $\chi_s$ . In Example 2, we illustrate how to apply our formal model in a real situation to find sequential interruptions, and the interruptional matrix.

**Example 2.** Table 3 is a list of five TE fragments from chromosome 1 position 448062 to 449273 taken from the output of RepeatMasker of Example 1. The five fragments belong to three TEs:  $X_1$ ,  $X_2$  and  $X_3$ , where the TE names of  $X_1, X_2, X_3$  are L1MD3, AluYc, AluSq.

**Table 3.** An example of sequential interruptions in chromosome 1

genoName	genoStart	genoEnd	genoLeft	strand	TEName	TEClass	TEStart	TEEnd	TELeft
chr1	448062	448139	-248802482	+	L1MD3	LINE	6988	7068	-814
chr1	448150	448328	-248802293	+	AluYc	SINE	122	299	0
chr1	448332	448403	-248802218	+	L1MD3	LINE	7068	7148	-847
chr1	448403	448710	-248801911	+	AluSq	SINE	1	313	0
chr1	448710	449273	-248801348	+	L1MD3	LINE	7149	7753	-242

Suppose the sequence of chromosome 1 is  $s$ , and the set of human TEs is  $\chi_s$ . The pruned sequence is  $\bar{s} = \beta_0 z_1 \beta_1 z_2 \beta_2 z_3 \beta_3 z_4 \beta_4 z_5 \beta_5$ , where  $\beta_0, \dots, \beta_5 \in \mathbb{N}$ , and  $z_1, z_3, z_5 \in \bar{X}_1, z_2 \in \bar{X}_2, z_4 \in \bar{X}_3, \bar{X}_1, \bar{X}_2, \bar{X}_3 \in \bar{\chi}(s \xrightarrow{RM} \chi_s)$ ,  $z_1 \stackrel{\epsilon, E}{\sim} z_3, z_3 \stackrel{\epsilon, E}{\sim} z_5$ . It is possible to see that there are two potential interruptions in  $s$ : an instance of  $X_1$  is present in the sequence, then an instance of  $X_2$  and an instance of  $X_3$  potentially inserted themselves into the instance of  $X_1$  to break it into three

segments  $z_1, z_3$  and  $z_5$ ; that is,  $|\Xi_s^{\epsilon,E}(X_2, X_1)| = 1$  and  $|\Xi_s^{\epsilon,E}(X_3, X_1)| = 1$ . Given a fixed order of the set of TEs as  $\chi_s = \{\dots, X_1, \dots, X_2, \dots, X_3, \dots\}$ , the interruption matrix showing only the rows and columns of these TEs is

$$M = \begin{bmatrix} \vdots & \vdots & \vdots \\ \dots 0 \dots 0 \dots 0 \dots \\ \vdots & \vdots & \vdots \\ \dots 1 \dots 0 \dots 0 \dots \\ \vdots & \vdots & \vdots \\ \dots 1 \dots 0 \dots 0 \dots \\ \vdots & \vdots & \vdots \end{bmatrix}. \tag{5}$$

From these sequential interruptions, using the approach from we can predict that the age of L1MD3 might be older than both AluYc and AluSq, but this provides no clue as to which one of AluYc and AluSq is older, because we do not know which of the two interruptions occurred first. By using the approach of [4] to rearrange this matrix, we can predict the TE ages.

Our notions in this section transformed the interruptional matrix construction described in prose in [4] into a formal model, which is more clear, and is easy to recreate.

### 4 Recursive Interruptions

When many insertions occurred throughout the evolution of a genomic sequence, the interruptions nest in a recursive pattern [10], which cannot be represented accurately with the interruptional matrix that only counts the abundance without storing the hierarchical relationships of interruptions. Indeed, Example 3 shows some nested TEs in real data.

**Example 3.** Table 4 is a list of TE fragments taken from the output of Repeat-Masker of Example 1. There are seven TE fragments starts from chromosome X position 53437061 to 53438226 that belong to four TEs:  $X_1, X_2, X_3$  and  $X_4$ , where the TE names of  $X_1, X_2, X_3, X_4$  are MIR, AluJb, AluSx, AluSq2.

**Table 4.** An example of recursive interruptions in chromosome X

genoName	genoStart	genoEnd	genoLeft	strand	TEName	TEClass	TEStart	TEEnd	TELeft
chrX	53437061	53437143	-101833417	+	MIR	SINE	3	88	-174
chrX	53437143	53437277	-101833283	+	AluJb	SINE	1	132	-170
chrX	53437277	53437448	-101833112	+	AluSx	SINE	39	192	-120
chrX	53437448	53437761	-101832799	+	AluSq2	SINE	1	312	0
chrX	53437761	53437887	-101832673	+	AluSx	SINE	193	312	0
chrX	53437887	53438055	-101832505	+	AluJb	SINE	133	293	-9
chrX	53438055	53438226	-101832334	+	MIR	SINE	89	261	-1

Suppose the sequence of chromosome  $X$  is  $s$ , and the set of human TEs is  $\chi_s$ . The pruned sequence is  $\bar{s} = \beta_0 z_1 \beta_1 z_2 \beta_2 z_3 \beta_3 z_4 \beta_4 z_5 \beta_5 z_6 \beta_6 z_7 \beta_7$ , where  $\beta_0, \dots, \beta_7 \in \mathbb{N}$ ,  $z_1, z_7 \in \bar{X}_1$ ,  $z_2, z_6 \in \bar{X}_2$ ,  $z_3, z_5 \in \bar{X}_3$ ,  $z_4 \in \bar{X}_4$ ,  $\bar{X}_1, \bar{X}_2, \bar{X}_3, \bar{X}_4 \in \bar{\chi}(s \xrightarrow{RM} \chi_s)$   $z_1 \stackrel{\varepsilon, E}{\sim} z_7$ ,  $z_2 \stackrel{\varepsilon, E}{\sim} z_6$ ,  $z_3 \stackrel{\varepsilon, E}{\sim} z_5$ . It is possible to see a potential process of nested interruptions described as: at first, an instance of *AluJb* inserted itself into an instance of *MIR* to break it into  $z_1$  and  $z_7$ ; then an instance of *AluSx* inserted itself into the instance of *AluJb* that has already presented in the sequence, to break it into  $z_2$  and  $z_6$ ; at the most recent time, an instance of *AluSq2* ( $z_4$ ) inserted itself into the presented *AluSx* instance to break it into  $z_3$  and  $z_5$ .

We can predict from the recursive interruptions that the age order of these three TEs from oldest to youngest might be: *MIR*, *AluJb*, *AluSx*, *AluSq2*.

The nested nature of the interruptions in Example 3 is not captured by the interruptional matrix, as the recursive nesting can “push” fragments so that they are no longer continuous. In this section we will first define a *recursive interruption context-free grammar* to model the generation of recursive interruptions, then discuss algorithms that calculate a parse tree of the grammar generating a given *order pruned sequence*, which shows a prediction of the hierarchical structure of TE insertions.

**Definition 10.** Given a set of TEs with a fixed order on its elements,  $\chi = \{X_1, X_2, \dots, X_m\}$ , the recursive interruption context-free grammar is a grammar  $G = (V, T, \delta, S)$ , where  $V = \{S, X_1, X_2, \dots, X_m\}$ ,  $T = \{1, 2, \dots, m\}$ , and  $\delta$  contains the following productions:

$$S \rightarrow X_i S, \quad 1 \leq i \leq m, \quad (1)$$

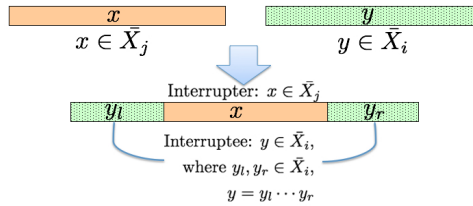
$$S \rightarrow X_i, \quad 1 \leq i \leq m, \quad (2)$$

$$X_i \rightarrow X_i X_j X_i, \quad 1 \leq i \leq m, \quad 1 \leq j \leq m, \quad (3)$$

$$X_i \rightarrow i, \quad 1 \leq i \leq m. \quad (4)$$

This grammar is used to generate strings over  $\{1, \dots, m\}^*$  corresponding to TE orders. Intuitively, productions of type (3) correspond to an instance of  $X_j$  inserting itself throughout evolution into an instance of  $X_i$ , as shown in Fig. 2, leaving a fragment from  $i$ , then  $j$ , then  $i$ . In a sentential form,  $X_i X_j X_i$  can either derive  $iji$  (using productions of type (4)) corresponding to that order of TEs, or any of them can be further interrupted (using productions of type (3)). Productions of type (1) correspond to independent positions of the sequence where a TE can insert itself (not a nested insertion, and can only be produced continuously from the root along the rightmost path of a parse tree). Productions of type (2) correspond to the final independent position of a TE insertion.

This context-free grammar is ambiguous. Indeed, it is clear that any string over  $T^+$  can be generated by  $G$  by using only productions of types (1), (2) and (4). This would require the application of  $2k$  productions to generate a string of length  $k$ . However, if there are  $l$  productions of type (3) applied, the total number of productions needed to generate a string of length  $k$  decreases to  $2(k - l)$ .

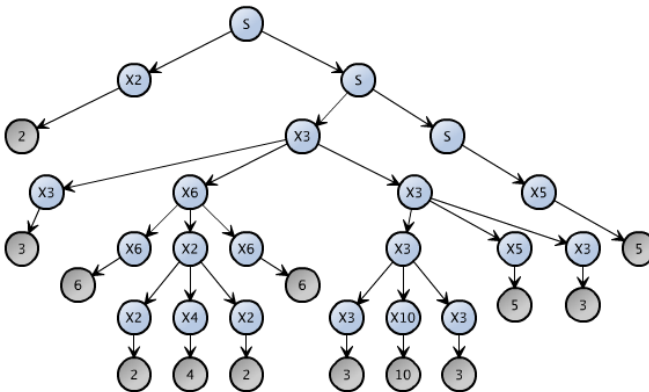


**Fig. 2.** A diagram showing an insertion of an instance of  $X_j$  inserting itself throughout evolution into an instance of  $X_i$

Since each production of type (1), (2), or (3) corresponds to one biological transposition, we are interested in parse trees which maximize the application of productions of type (3), or minimize the total number of productions applied. This would correspond to minimizing the number of transpositions that occurred throughout evolution.

Example 4 shows how nested interruptions in a sequence are generated by the grammar as the yield of its one possible parse tree that maximized the application of productions of type (3).

**Example 4.** Given a genomic sequence  $s$  and a set of TEs with a fixed order on its elements  $\chi_s = \{X_1, X_2, \dots, X_{10}\}$ , and assume  $s = w_0 z_1 w_1 \dots z_{13} w_{13}$ , as in Equation (2) with  $z_1, z_4, z_6 \in \bar{X}_2$ ,  $z_2, z_8, z_{10}, z_{12} \in \bar{X}_3$ ,  $z_5 \in \bar{X}_4$ ,  $z_{11}, z_{13} \in \bar{X}_5$ ,  $z_3, z_7 \in \bar{X}_6$ ,  $z_9 \in \bar{X}_{10}$ . Then an order pruned sequence  $\bar{s}_o = 2\ 3\ 6\ 2\ 4\ 2\ 6\ 3\ 10\ 3\ 5\ 3\ 5$  is the yield of the parse tree shown in Fig. 3.



**Fig. 3.** A parse tree of  $G$  from Definition 10 that yields  $\bar{s}_o$

The recursive interruption context-free grammar is a very simple and general way of capturing the recursive nature of TE interruptions. However, the order pruned sequence generated by the grammar only contains the TEs (names) to



which the detected TE fragments belong. It does not take into account where each fragment lies within a TE, and there is neither a way to determine the actual distance where the two fragments lie with respect to the genomic sequence. We leave this direction as future work.

As discussed, given an order pruned sequence, we are interested in finding a parse tree of the grammar that maximizes the applications of productions of type (3), or minimizes the overall productions applied to generate this sequence. A stochastic context-free grammar is a context-free grammar, where every production has an associated probability value between 0 and 1, such that the probability for all productions on a nonterminal adds to 1. The probability associated with a parse tree is the product of the probabilities of the production instances applied to produce it.

Considering the grammar in Definition 10, since all probabilities are between 0 and 1, trees that use fewer productions will tend to have a higher probability. A *most likely parse tree*, defined as a parse tree with the highest probability, corresponds to the parse tree that has the most productions of type (3) applied in the recursive interruption context-free grammar. For this grammar, if we give all productions for each nonterminal equal weight (for each production of  $X_i$ , the probability is  $1/(m + 1)$ , and for each production of  $S$ , the probability is  $1/(2m)$ ), the CYK algorithm [3] can find a most likely parse tree that has a given sequence as yield. In our case, starting with the order pruned sequence, it can predict a most likely parse tree with it as the yield.

The complexity of CYK algorithm is  $O(L^3M^3)$  [3], where  $L$  is the length of the order pruned sequence (corresponding to the number of TE fragments detected in a genomic sequence), and  $M$  is the number of nonterminals in the grammar (corresponding to the total number of transposons of that organism in Repbase Update plus one), which will be very lengthy in practice. We leave as future work an investigation of algorithms that can be faster while taking additional positional information into account in generating the most likely parse tree.

In the future, we will work on algorithms for detecting the sequential and recursive interruptions, which will be potentially used to analyze the interruptional patterns of a given genomic sequence and generate the interruption matrix about TE interruptions, whose results are useful in further modelling and analysis. We will also work on the algorithms generating a most likely parse tree by taking extra information into account, and using heuristics for speedups.

## References

1. Batzer, M., Deininger, P., Hellmann-Blumberg, U., Jurka, J., Labuda, D., Rubin, C., Schmid, C., Zietkiewicz, E., Zuckerkandl, E.: Standardized Nomenclature for Alu Repeats. *Journal of Molecular Evolution* 42(1), 3–6 (1996)
2. Belancio, V., Roy-Engel, A., Deininger, P.: All y'all need to know 'bout retroelements in cancer. In: *Seminars in Cancer Biology*, vol. 20, pp. 200–210. Elsevier (2010)
3. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge (1998)

4. Giordano, J., Ge, Y., Gelfand, Y., Abrusán, G., Benson, G., Warburton, P.: Evolutionary History of Mammalian Transposons Determined by Genome-wide Defragmentation. *PLoS Computational Biology* 3(7), e137 (2007)
5. Gregory, T.: *The Evolution of the Genome*. Academic Press (2005)
6. Hopcroft, J.E.: *Introduction to Automata Theory, Languages, and Computation*, 3/E. Pearson Education India (2008)
7. Jones, N.C., Pevzner, P.: *An Introduction to Bioinformatics Algorithms*. MIT press (2004)
8. Jurka, J., Kapitonov, V., Pavlicek, A., Klonowski, P., Kohany, O., Walichiewicz, J.: Repbase Update, a database of eukaryotic repetitive elements. *Cytogenetic and Genome Research* 110(1-4), 462–467 (2005)
9. Kapitonov, V., Jurkal, J.: The Age of Alu Subfamilies. *Journal of Molecular Evolution* 42(1), 59–65 (1996)
10. Kronmiller, B.A., Wise, R.P.: TEneSt: Automated Chronological Annotation and Visualization of Nested Plant Transposable Elements. *Plant Physiology* 146(1), 45–59 (2008)
11. Lander, E., Linton, L., Birren, B., Nusbaum, C., Zody, M., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., et al.: Initial Sequencing and Analysis of the Human Genome. *Nature* 409(6822), 860–921 (2001)
12. Lerat, E.: Identifying Repeats and Transposable Elements in Sequenced Genomes: How to Find Your Way through the Dense Forest of Programs. *Heredity* 104(6), 520–533 (2009)
13. Smit, A., Toth, G., Riggs, A., Jurka, J.: Ancestral, Mammalian-wide Subfamilies of LINE-1 Repetitive Sequences. *Journal of Molecular Biology* 246(3), 401–417 (1995)
14. Smit, A.F.A., Hubley, R., Green, P.: RepeatMasker Open-3.0 (1996-2010), <http://www.repeatmasker.org>

# Probabilistic Analysis of Long-Term Swarm Performance under Spatial Interferences

Yara Khaluf<sup>1</sup>, Mauro Birattari<sup>2</sup>, and Franz Rammig<sup>1</sup>

<sup>1</sup> Heinz Nixdorf Institut, University of Paderborn  
Fürstenallee 11, 33102 Paderborn, Germany  
{yara, franz}@hni.uni-paderborn.de  
<https://www.hni.uni-paderborn.de/>

<sup>2</sup> IRIDIA, Université Libre de Bruxelles  
B-1050 Brussels, Belgium  
mbiro@ulb.ac.be  
<http://iridia.ulb.ac.be>

**Abstract.** Swarm robotics is a branch of collective robotics that outperforms many other systems due to its large number of robots. It allows for performing several tasks that are beyond the capability of a single or multi robot systems. Its global behaviour emerges from the local rules implemented on the level of its individual robots. Thus, estimating the obtained performance in a self-organized manner represents one of the main challenges, especially under complex dynamics like spatial interferences. In this paper, we exploit the central limit theorem (CLT) to analyse and estimate the swarm performance over long-term deadlines and under potential spatial interferences. The developed model is tested on the well-known foraging task, however, it can be generalized to be applied on any constrictive robotic task.

**Keywords:** Swarm robotics, Time-constrained tasks, Central limit theorem.

## 1 Introduction

Swarm robotics is a high density multi-robot system, where the global behaviour emerges from local rules implemented on the level of individual robots. These systems are characterized by a set of advantages including: redundancy, scalability and flexibility which introduce them as a promising approach for a large spectrum of tasks.

Spatial interferences, on the other hand, affect significantly the performance of the single robot and consequently the collective performance of the swarm. A well-studied example is the foraging, where robots are exploited to retrieve scattered objects to a special area called "nest". As noted in [5,10], the increment of the robots' number in a task like foraging, decreases the performance of a single robot which represents the number of retrieved objects per time unit. In the case of swarm performance, it may increase by adding robots up to an

optimal number and after that it starts to decrease affected by the interferences among robots. Although the influence of spatial interferences among robots has been studied on a limited number of swarm scenarios, mainly on foraging, the observations do not appear to be surprising. Increasing the number of robots, increases the time required by the single robot to accomplish individual parts of the task because of the concurrence among robots. This decreases in turn, the number of parts that could be accomplished by single robots within a specific time period. From the swarm point of view, the obtained performance keeps increasing as long as the benefit of parallelising the work is still larger than the time penalty paid because of the interferences. Swarm robotic systems are prone to intensive spatial interferences and as robotic missions are generally characterized by their long-term durations, providing the ability to estimate the swarm performance for long-term tasks and under the spatial interferences is of a significant importance. However, performing this estimation by running real experiments or computer simulations is intensive time and resource consuming. Consequently, alternative tools are required to perform such kind of estimations. In this paper, we investigate the use of the Central Limit Theorem (CLT) as a tool for approximating the swarm performance and analysing it probabilistically over long-term deadlines and under robots spatial interferences. Central limit theorem, in its classic version, states that the mean of a sufficiently large set of independent and identically distributed random variables each with a finite mean and variance tends to be distributed normally. How large the set of i.i.d. should be, is based on their distribution parameters.

The rest of the paper is organized as following: Section 2 lists a set of related work. Section 3 formulates the problem of the swarm performance estimation over long-term deadlines. In Section 4 The central limited theorem is introduced and the proposed estimator model for the swarm performance is illustrated. Section 5 presented a foraging scenario to verify the proposed estimation strategy and Section 6 concludes the paper.

## 2 Related Work

Swarm robotic performance is influenced by the interferences among participating robots [2]. Most of the performed studies were focusing on characterizing how the amount of work accomplished within a specific time unit, changes by changing the number of working robots. The studies were mostly accomplished on well-known swarm tasks like foraging and the conclusions were similar, namely, that increasing the number of robots decreases the performance of individual robots. After characterizing the relation between the swarm size and performance, several studies were performed to improve the swarm performance by reducing the density of spatial interferences. In [1] several types of interferences in multi-robot systems have been defined and it presented the interactions among robots working together in a common area, like the nest, as the main type of robots interactions. The authors have proposed two techniques to arbitrate the impact of interactions. First, by making sure that robots are working in different

areas and second, by scheduling the occupation of shared areas. The first proposal was further investigated under the term *bucket-brigade* like in [15], [16] and [10], in addition to [4], where the approach was extended to consider adaptive working areas. Task partitioning represents another technique, which is used to improve the swarm performance under spatial interferences. In [12] a task partitioning technique was proposed, in which the shared area was divided into two areas and the robots select their area using a threshold mechanism. In [13], the authors studied the role of task partitioning in reducing the concurrent access to the nest area in a harvesting task.

On the other hand, characterizing swarm performance by means of real experiments is not always possible and is an expensive solution from both time and hardware points of view. In addition, computer simulations are very time consuming especially when tasks are associated with long-term deadlines. In such cases, the mathematical modelling represents one of the best approaches. A mathematical model has been introduced in [5], which characterizes the performance of the single robot and the swarm under spatial interferences. In [9] a list of the various mathematical models which can be applied in swarm systems, is reported. Most of these mathematical studies were focusing on specific swarm scenarios like foraging in [7] or collaborative distributed manipulation in [8]. To our best knowledge no study was focusing on the mathematical analysis of the collective swarm performance within specific deadlines. In addition, the probability analysis of swarm robotics was not considered intensively and only few studies were performed in that field like in [6]. The central limit theorem [14], is a wide-applied theorem in many fields related to measurement approximations and hypothesis testing. However, it is not investigated yet within the context of swarm robotics. This important theorem is exploited, here, to develop swarm performance estimators which allow for a probabilistic characterization of swarm performance over long-term deadlines and under the dynamics of spatial interferences.

### 3 Problem Formulation

We consider constructive tasks where the total contribution on the task within a particular time period is the sum of the individual contributions of robots over that time period. Each of the considered tasks can be characterized by its long-term deadline, that represents the time point after which the robots should stop to work on the task.

In swarm robotic systems the contribution of a single robot on any task is a random variable which can belong to the discrete, as well as, to the continuous space based on the type of the task. In a task like pushing a box, the robot performance is a continuous random variable represents the distance the box travels within a specific time unit. However, in a foraging task the robot performance represents the number of retrieved objects and belongs to the discrete space. In this paper, we focus on the discrete space of robot performance where the task consists of discrete parts to be accomplished.

Let us assume a task  $T_i$  with the deadline  $D_i$  and a homogeneous swarm of  $N$  simple robots. Each of these robots is able to accomplish one part of  $T_i$

at a time. We use  $\beta_{ij}(D_i)$  to denote the discrete random variable associated with the number of parts can be accomplished by the robot  $R_j$  on task  $T_i$  up to its deadline of  $D_i$ , under the influence of spatial interferences. The swarm performance on task  $T_i$  up to the deadline  $D_i$  is denoted by  $\omega_i(D_i)$ . Thus,  $\omega_i(D_i)$  is the total number of parts accomplished by the swarm within the time of  $D_i$  under the spatial interferences and is calculated as the sum of the robots' individual contributions:

$$\begin{aligned}\omega_i(D_i) &= \beta_{i1}(D_i) + \beta_{i2}(D_i) + \dots + \beta_{iN}(D_i) \\ &= \sum_{j=1}^N \beta_{ij}(D_i)\end{aligned}\quad (1)$$

We divide the time period between the start of the execution  $t = 0$  and the task deadline  $D_i$  into equal and non-overlapping time-windows each with the length  $\tau$ . The length  $\tau$  of the time-window is selected under the following constraints: It should be equal to or greater than the average time required by a single robot to accomplish one part on task  $T_i$ , The task deadline  $D_i$  should be a multiplier of  $\tau$  and  $\tau$  should be significantly smaller than the task deadline:  $\tau \ll D_i$ .

The swarm performance at deadline  $D_i$  is the sum of the *swarm* contributions over all the time-windows included within the deadline  $D_i$ . Hence, we can calculate the swarm performance at the deadline  $D_i$  as in following:

$$\begin{aligned}\omega_i(D_i) &= \omega_i(\tau_1) + \omega_i(\tau_2) + \dots + \omega_i(\tau_K) \\ &= \sum_{j=1}^K \omega_i(\tau_j)\end{aligned}\quad (2)$$

where  $K$  is the number of time-windows included in deadline  $D_i$ .

On the other hand, the swarm performance at deadline  $D_i$  is the sum of the *individual* robots' contributions over all the time-windows included in the deadline  $D_i$ . Using Equation (1) the swarm performance can be calculated in terms of the individual robots' contributions as in following:

$$\begin{aligned}\omega_i(D_i) &= (\beta_{i1}(\tau_1) + \beta_{i2}(\tau_1) + \dots + \beta_{iN}(\tau_1)) \\ &\quad + \dots \\ &\quad + (\beta_{i1}(\tau_K) + \beta_{i2}(\tau_K) + \dots + \beta_{iN}(\tau_K)) \\ &= \sum_{j=1}^K \sum_{l=1}^N \beta_{il}(\tau_j)\end{aligned}\quad (3)$$

The goal is to estimate the performance which can be obtained by a swarm of  $N$  robot at the deadline  $D_i$  and under the influence of spatial interferences. We aim to perform this estimation with the minimum time and resources consumption by launching short-time real experiments or computer simulations. The estimation is carried out in a probabilistic manner, where we derive the probability density function (PDF) in addition to the cumulative distribution function (CDF) of

the random variable that is associated with the swarm performance. Such a probabilistic analysis helps us to answer questions like: "What is the probability of achieving a specific swarm performance  $S_i$  within the deadline  $D_i$  under the influence of spatial interferences?"

$$\Pr(\omega_i(D_i) \geq S_i) \quad (4)$$

## 4 Probabilistic Analysis of Swarm Performance under Spatial Interferences

In this section, we investigate the well-known central limit theorem in performing a probabilistic analysis of the swarm performance at long-term deadlines under the influence of spatial interferences. Central limit theorem is a wide-applied theorem specially in fields of hypothesis testing, cancelling of communication noise and statistic [3]. Let us have a set of  $n$  independent and identically distributed random variables  $\{X_1, X_2, \dots, X_n\}$ , sampled from a distribution with a specific mean  $\mu$  and a variance  $\sigma^2$ . The central limit theorem (CLT) in its classic version states that for sufficiently large  $n$ , the sum of the  $n$  random variables is normally distributed and can be characterized with the following mean and variance:

$$\mu_n = n\mu \quad (5) \qquad \sigma_n^2 = n\sigma^2 \quad (6)$$

The swarm contribution  $\omega_i(\tau_j)$  over the time-window  $\tau_j$ , is the random number of parts accomplished by the swarm on task  $T_i$  within the time-window  $\tau_j$ . Concurrently, the single robot contribution  $\beta_{it}(\tau_j)$  is the random number of parts accomplished by a single robot on task  $T_i$  within the time-window  $\tau_j$ . The mean and the variance associated with these two random variables are influenced directly by the number of robots working on the task in addition to the work density available on this task. The work density here refers to the number of parts available on the task and need to be accomplished by the robots. Swarm systems are generally characterized by their large sizes, thus the failure of a few robots or the addition of another few, will not apply a considerable change on the average performance of the swarm or of the single robot. In addition applying the central limit theorem is constrained by having a sufficiently large  $n$ , where  $n$  represents the number of robots in case the swarm performance is defined based on Equation (3). Hence, a small change in the number of robots will not change significantly the total performance of the swarm. On the other hand, the assumption of having a constant work density is associated with a large set of real-world applications. Examples to these tasks could be recycling systems where the robots are responsible to retrieve objects excreted *continuously* at specific locations to some recycling destination. Another example could be a production-transport system, where objects are assumed to be produced *continuously* at different locations and require to be transported to specific delivery points.

Based on the above discussion we can describe the two conditions of having a constant average of robots number and a constant work density as realistic assumptions. Let us denote the mean and the standard deviation of the random variable associated with the swarm performance  $\omega_i(\tau)$  on task  $T_i$  within the

time-window  $\tau$  respectively by:  $\mu_{\omega_i(\tau)}$  and  $\sigma_{\omega_i(\tau)}$ . Based on the central limit theorem the long-term swarm performance will be normally distributed:

$$\omega_i(D_i) \sim Norm(K\mu_{\omega_i(\tau)}, K\sigma_{\omega_i(\tau)}^2) \quad (7)$$

On the other hand, we denote the mean and the standard deviation of the random variable associated with the single robot performance  $\beta_{ij}(\tau)$  respectively by:  $\mu_{\beta_{ij}(\tau)}$  and  $\sigma_{\beta_{ij}(\tau)}$ . According to the central limit theorem, the long-term swarm performance will be also normally distributed:

$$\omega_i(D_i) \sim Norm(KN\mu_{\beta_{ij}(\tau)}, KN\sigma_{\beta_{ij}(\tau)}^2) \quad (8)$$

Consequently, the swarm performance can be characterized probabilistically by using the cumulative distributed function (CDF) of the normal distribution, which for the mean  $\mu$  and the variance  $\sigma^2$ , is defined as in following:

$$\Pr(X \leq x) = \frac{1}{2} + \frac{1}{2} \operatorname{erf}\left(\frac{x - \mu}{\sqrt{2\sigma^2}}\right) \quad (9)$$

We substitute the random variable  $X$  by the swarm performance  $\omega_i(D_i)$  and the value of small  $x$  by a desired performance  $S_i$ , that represents the number of parts to accomplish up to the task deadline  $D_i$ . The probability we are interested to calculate is the one in Equation (4):  $\Pr(\omega_i(D_i) \geq S_i)$ , which can be expressed using the CDF of the normal distribution (9):

$$\begin{aligned} \Pr(\omega_i(D_i) \geq S_i) &= \Pr(\omega_i(D_i) > (S_i - 1)) \\ &= 1 - \Pr(\omega_i(D_i) \leq (S_i - 1)) \end{aligned} \quad (10)$$

Consequently, the central limit theorem (CLT) can be applied to estimate the long-term performance of swarm robotics, efficiently, in both following contexts:

- Swarm performance over short experiments: the swarm performance at deadline  $D_i$  is the sum of the *swarm* contributions over all time-windows included within the deadline  $D_i$ . We map each of these swarm contributions to a random variable with the mean  $\mu_{\omega_i(\tau)}$  and the variance  $\sigma_{\omega_i(\tau)}^2$  which are measured over one experiment of the length  $\tau$ . Consequently, the central limit theorem can be applied to approximate the swarm performance at  $D_i$  as the sum of these random variables, like in Equation (2).
- Single robot performance over short experiments: the swarm performance at deadline  $D_i$  can be calculated as the sum of the *individual* robots' contributions over all time-windows included within the deadline  $D_i$ . As the single robot performance over one time-window experiment can be measured by the robot itself, this estimation technique represents a "self-organized" one. The robot works on the task for one time-window to estimate the mean and the standard deviation of the random time required to accomplish one part of that task. After that, the central limit theorem is applied to approximate the swarm performance at  $D_i$  using Equation (3).



## 5 Scenario and Evaluation

We consider the foraging scenario, where a large number of objects are scattered uniformly over an object-area and need to be retrieved by a swarm of robots to some nest-area. During the experiment, each robot can be in one of the following states: *exploring* or *retrieving*. A robot being in the exploring state searches for objects to retrieve and as soon as an object is found, the robot changes its state to retrieving and starts to move towards the nest-area. It is assumed that new objects are popping up in the arena and the objects density remains constant throughout the whole experiment. The ARGoS<sup>1</sup> simulator [11] is used to calculate an average performance function via repeated high-level simulations in order to characterize the effect of the spatial interferences on the performance of the single robot as well as on the swarm. The simulations are repeated for 125 times. Figure 1 shows how the mean of the single robot performance decreases by increasing the number of robots in addition to the standard deviation of this performance for different swarm sizes. Figure 2 shows the change in the mean of the swarm performance and its standard deviation while applying the same increment in the swarm size. We consider the foraging scenario illustrated in Section 5, where it is carried out by a swarm of 30 robots within the deadline  $D_i = 12 \cdot 10^3$  seconds. The time-window length is set to  $\tau = 100$ . Figure 3 shows the mean  $\mu$  in addition to the  $3 \times \sigma$  of the random number associated with the retrieved objects over all time-windows up to the  $D_i = 12 \cdot 10^3$  seconds. Figure 4 shows the time it takes the average of the system performance to stabilize. This time is referred to as *start-up time*. At the beginning of the foraging task all robots are free to search for objects and to retrieve them as soon as they find any. Thus, the number of objects retrieved at the beginning is higher than the number will be retrieved later on, when the robots are divided between robots which are searching and free to retrieve and robots which are retrieving. This is the reason behind the existence of such a start-up time after which the system performance stabilizes. The accuracy of the CLT estimation of the swarm performance is influenced by including the system performance during the start-up time or excluding it. This influence varies based on the relative relation between the length of both: the deadline  $D_i$  and the start-up time. The swarm performance will be estimated within the two contexts mentioned above. First by using the swarm contributions and second using the individual robot's contribution both measured over short experiments.

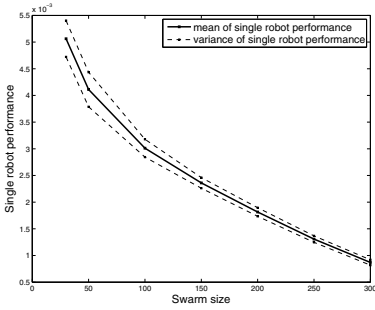
### – Swarm performance over short experiments:

We use the swarm contribution achieved by the whole swarm within one time-window  $\tau = 100$  second. We substitute it in Equation (2), where the deadline of  $D_i = 12 \cdot 10^3$  includes 120 time-window of the length 100 seconds:

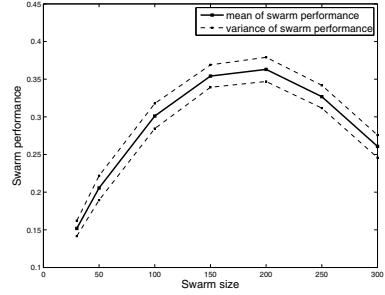
$$\omega_i(D_i) = \sum_{j=1}^K \omega_i(\tau_j) \quad \Rightarrow \quad \omega_i(12 \cdot 10^3) = \sum_{j=1}^{120} \omega_i(100) \quad (11)$$

---

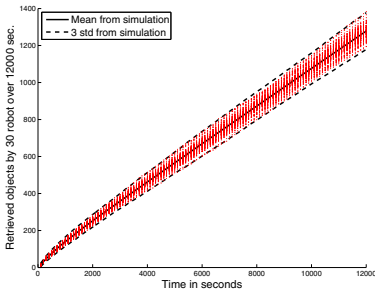
<sup>1</sup> ARGoS is a discrete-time physics-based simulation framework developed within the Swarmanoid project. It can simulate various robots at different levels of details, as well as a large set of sensors and actuators.



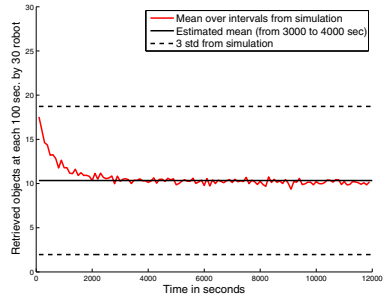
**Fig. 1.** Single robot performance under spatial interferences during 1 second



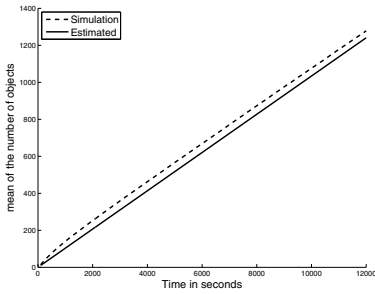
**Fig. 2.** Swarm performance under spatial interferences during 1 second



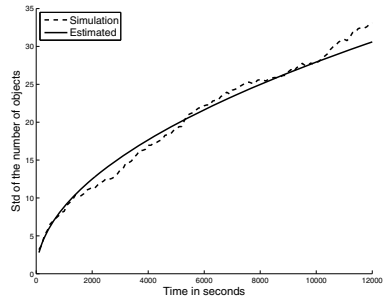
**Fig. 3.**  $\mu$  and  $3 \times \sigma$  of the number of objects retrieved during  $12 \cdot 10^3$  seconds



**Fig. 4.**  $\mu$  and  $3 \times \sigma$  of the number of objects retrieved over all 100 seconds intervals included in the deadline  $D_i$



**Fig. 5.** Mean of retrieved objects throughout  $2 \cdot 10^3$  seconds vs. the mean predicted by CLT



**Fig. 6.** Standard deviation of retrieved objects throughout  $2 \cdot 10^3$  seconds vs. the mean predicted by CLT

According to the central limit theorem, the r.v. associated with the number of retrieved objects at the deadline  $12 \cdot 10^3$  is normally distributed with the mean and the standard deviation as in following:

$$\mu_{\omega_i(12 \cdot 10^3)} = 120\mu_{\omega_i(100)} \quad (12) \quad \sigma_{\omega_i(12 \cdot 10^3)} = \sqrt{120}\sigma_{\omega_i(100)} \quad (13)$$

Figures 5 and 6 illustrate the mean and the standard deviation of the number of objects retrieved within the deadline  $D_i = 2 \cdot 10^3$ , compared to the mean and standard deviation predicted by CLT using 12 and 13.

The probability of retrieving more than  $S_i$  objects within the deadline  $D_i$  can be derived using the CDF of the normal distribution using in Equation (9):

$$\begin{aligned} \Pr(\omega_i(2 \cdot 10^3) \geq S_i) &= 1 - \left[ \frac{1}{2} + \frac{1}{2} \operatorname{erf} \left( \frac{(S_i - 1) - \mu_{\omega_i(2 \cdot 10^3)}}{\sqrt{2}\sigma_{\omega_i(2 \cdot 10^3)}} \right) \right] \\ &= 1 - \left[ \frac{1}{2} + \frac{1}{2} \operatorname{erf} \left( \frac{(S_i - 1) - 120 \mu_{\omega_i(100)}}{\sqrt{2} \sqrt{120} \sigma_{\omega_i(100)}} \right) \right] \end{aligned} \quad (14)$$

By performing 125 ARGoS simulations, Figures 7 and 8 show the probability density function (PDF) associated with the number of objects retrieved by the swarm at deadline  $D_i = 2 \cdot 10^3$ . In Figure 7 using the mean  $\mu_{\omega_i(100)} = 10.3411$  and standard deviation  $\sigma_{\omega_i(100)} = 2.7931$  measured after the system stabilizes and in Figure 8 using the the mean  $\mu_{\omega_i(100)} = 10.6239$  and standard deviation is  $\sigma_{\omega_i(100)} = 2.9755$  measured with taking the system performance though the start-up time into account. Figures 9 and 10 show the cumulative distribution function (CDF) associated with the number of retrieved objects also in Figure 9 after the system stabilizes and in Figure 10 with taking the start-up time into account.

#### – Single robot performance over short experiments:

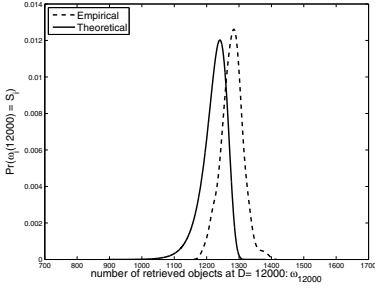
Here, we estimate the swarm performance at deadline  $D_i = 2 \cdot 10^3$  in a self-organized way by using the single robot performance measured within one time-window by substituting it in Equation (3).

$$\omega_i(D_i) = \sum_{j=1}^K \sum_{l=1}^N \beta_{il}(\tau_j) \quad \Rightarrow \quad \omega_i(12 \cdot 10^3) = \sum_{j=1}^{120} \sum_{l=1}^{30} \beta_{il}(100) \quad (15)$$

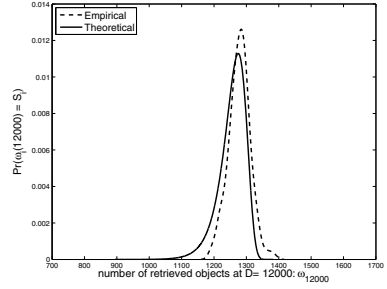
According to the CLT, the r.v. associated with objects retrieved by the swarm up to the deadline  $12 \cdot 10^3$ , is normally distributed with the mean and the standard deviation as in following:

$$\mu_{\omega_i(12 \cdot 10^3)} = 120 \times 30 \mu_{\beta_{ij}(100)} \quad (16) \quad \sigma_{\omega_i(12 \cdot 10^3)} = \sqrt{120 \times 30} \sigma_{\beta_{ij}(100)} \quad (17)$$

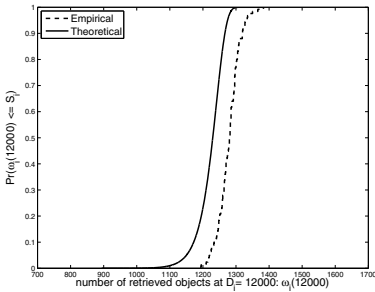
This estimation can be performed by the robots themselves and could help them in making appropriate allocation decisions. The probability of interest in Equation (4) can be calculated by applying the CDF of normal distribution (9):



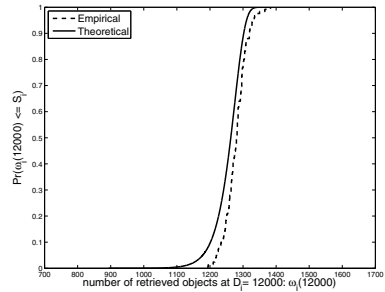
**Fig. 7.** PDF of the retrieved number of objects without taking the start-up time into account



**Fig. 8.** PDF of the retrieved number of objects with taking the start-up time into account



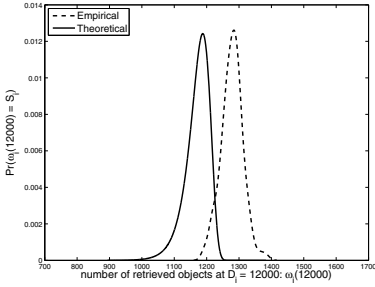
**Fig. 9.** CDF of the retrieved number of objects without taking the start-up time into account



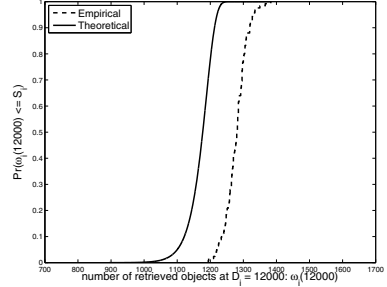
**Fig. 10.** CDF of the retrieved number of objects with taking the start-up time into account

$$\begin{aligned}
 \Pr(\omega_i(2 \cdot 10^3) \geq S_i) &= 1 - \left( \frac{1}{2} + \frac{1}{2} \operatorname{erf} \left( \frac{(S_i - 1) - \mu_{\omega_i}(2 \cdot 10^3)}{\sqrt{2} \sigma_{\omega_i}(2 \cdot 10^3)} \right) \right) \\
 &= 1 - \left( \frac{1}{2} + \frac{1}{2} \operatorname{erf} \left( \frac{(S_i - 1) - 120 \times 30 \mu_{\beta_{i,j}}(100)}{\sqrt{2} \sqrt{120 \times 30} \sigma_{\beta_{i,j}}(100)} \right) \right) \quad (18)
 \end{aligned}$$

The swarm performance here is simulated only in the situation of taking the start-up time into account. Measuring the swarm performance after the system stabilizes is straightforward as mentioned above. Figure 11 shows the comparison between the estimated (PDF) of the r.v. associated with the number of objects retrieved by the 30 robot at  $D_i = 2 \cdot 10^3$  and the empirical one. Figure 12 shows the same comparison but for the cumulative distribution function(CDF) of both. The reason why the estimation performed using the individual robot's contribution is not the same accurate as the one performed using the swarm contribution, is that, in the case of the swarm the average performance of  $N$  robot is taken into account.



**Fig. 11.** PDF of the number of retrieved objects estimated based on single robot performance



**Fig. 12.** CDF of the number of retrieved objects estimated based on single robot performance

## 6 Conclusion

In this paper, we presented a probabilistic study of the swarm performance that can be achieved within long-term deadlines and under the influence of spatial interferences. Estimating the performance of robots swarms is an important concept especially for tasks where the performance should be planned under specific constraints like time constraints. In such cases the early estimation of the performance will be obtained at the long-term deadline of the considered task, is of a significant importance.

The central limit theorem CLT, is a straightforward tool and the core theorem which was investigated in this paper to perform the global performance estimation of the swarm. Such a mathematical estimation represents a useful tool to preserve time and resources in comparison to real-time experiments or computer simulations. The estimation of the swarm robotic performance over long-term deadlines using the central limit theorem was accomplished into two contexts: first by using the swarm contributions over short-term experiments and second in a self-organized way by using the single robot performance over short experiments.

The accomplishment of the performance estimation over short experiments, allows for the possibility of launching repair mechanisms at an early stage of the execution. In addition, this estimation is considered to be useful especially in cases where the swarm performance and/or the single robot performance are not following a well-known distribution like the normal distribution. In such cases, the central limit theorem can be applied efficiently to accomplish swarm performance estimations.

## References

1. Goldberg, D.: Evaluating the dynamics of agent-environment interaction. Ph.D. thesis, University of Southern California (2001)
2. Goldberg, D., Matarić, M.J.: Reward maximization in a non-stationary mobile robot environment. In: Proceedings of the fourth International Conference on Autonomous Agents, AGENTS 2000, pp. 92–99. ACM, New York (2000)

3. Jacod, J., Podolskij, M., Vetter, M.: Limit theorems for moving averages of discretized processes plus noise. *The Annals of Statistics* 38(3), 1478–1545 (2010)
4. Lein, A., Vaughan, R.T.: Adaptive multirobot bucket brigade foraging. In: *Proceedings of the Eleventh International Conference on Artificial Life, ALife XI*, pp. 337–342. MIT Press (2008)
5. Lerman, K., Galstyan, A.: Mathematical model of foraging in a group of robots: Effect of interference. *Autonomous Robots* 13(2), 127–141 (2002)
6. Lerman, K., Martinoli, A., Galstyan, A.: A Review of Probabilistic Macroscopic Models for Swarm Robotic Systems. In: Şahin, E., Spears, W.M. (eds.) *Swarm Robotics 2004*. LNCS, vol. 3342, pp. 143–152. Springer, Heidelberg (2005)
7. Liu, W., Winfield, A.F.T., Sa, J.: Modelling swarm robotic systems: A case study in collective foraging. In: *Towards Autonomous Robotic Systems (TAROS 2007)*, pp. 25–32. Aberystwyth (2007)
8. Martinoli, A., Easton, K., Agassounon, W.: Modeling swarm robotic systems: A case study in collaborative distributed manipulation. *Int. Journal of Robotics Research* 23, 415–436 (2004)
9. Muniganti, P., Pujol, A.O.: A survey on mathematical models of swarm robotics. In: *Workshop of Physical Agents* (2010)
10. Ostergaard, E.H., Sukhatme, G.S., Mataric, M.J.: Emergent bucket brigading: a simple mechanism for improving performance in multi-robot constrained-space foraging tasks. In: *Proceedings of the Fifth International Conference on Autonomous Agents, AGENTS 2001*, pp. 29–30. ACM, New York (2001)
11. Pinciroli, C., Trianni, V., OGrady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Caro, G., Ducatelle, F., Birattari, M., Gambardella, L., Dorigo, M.: Argos: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence* 6, 271–295 (2012)
12. Pini, G., Brutschy, A., Birattari, M., Dorigo, M.: Interference reduction through task partitioning in a robotic swarm or: dont you step on my blue suede shoes! (2009)
13. Pini, G., Brutschy, A., Birattari, M., Dorigo, M.: Task partitioning in swarms of robots: Reducing performance losses due to interference at shared resources. In: Cetto, J., Filipe, J., Ferrier, J.L. (eds.) *Informatics in Control Automation and Robotics*. LNEE, vol. 85, pp. 217–228. Springer, Heidelberg (2011)
14. Rice, J.A.: *Mathematical Statistics and Data Analysis*, 3rd edn. Duxbury Press (April 2001)
15. Shell, D.A., Mataric, M.J.: On foraging strategies for large-scale multi-robot systems. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2717–2723. IEEE (2006)
16. Vaughan, A.L.R.T.: Adaptive multi-robot bucket brigade foraging. *Artificial Life* 11, 337 (2008)

# Improving MLP Neural Network Performance by Noise Reduction

Mirosław Kordos<sup>1</sup> and Andrzej Rusiecki<sup>2</sup>

<sup>1</sup> University of Bielsko-Biala, Department of Mathematics and Computer Science  
Bielsko-Biala, Willowa 2, Poland  
`mkordos@ath.bielsko.pl`

<sup>2</sup> Wrocław University of Technology  
Institute of Computer Engineering, Control and Robotics  
Wrocław, Wybrzeże Wyspińskiego 27, Poland  
`andrzej.rusiecki@pwr.wroc.pl`

**Abstract.** In this paper we examine several methods for improving the performance of MLP neural networks by eliminating the influence of outliers and compare them experimentally on several classification and regression tasks. The examined methods include: pre-training outlier elimination, use of different error measures during network training, replacing the weighted input sum with weighted median in the neuron input functions and various combinations of them. We show how these methods influence the network prediction. Based on the experimental results, we also present a novel hybrid approach improving the network performance.

## 1 Introduction

An outlier is an example that is numerically distant from the rest of the surrounding data. That can be either a point that is close to its neighbors in the input space, but distant from the output space (different class or much different value in the case of regression) or that is far from any points as well in the input as in the output space. Outliers often indicate either of measurement error or some data points that are so rare that should not be taken into account while building the data model. Thus we want either to discard them or use approaches that are robust to outliers. Another problem is that sometimes it cannot be clearly stated if a given point is an outlier or not and rather some degree of being an outlier that a crisp decision is preferred. In that case the model does not entirely disregard such a point, but decreases its influence on the model parameters.

Multilayer perceptron neural networks (MLP) are one of the models that can be used to represent the data. They are trained by minimizing an error function on the training set, to make the network map the input data distribution to the output space variable, which can be either discrete in the case of classification or continuous in the case of regression, or can represent some structured data. The performance of the trained network obviously depends not only on the network architecture and learning algorithms, but on the quality of the training data as well. A noisy dataset with many outliers does not describe well the desired

mapping from the input to output space. In this case also the neural network trained on that data will not implement the proper mapping.

The methods designed to make the network training robust to gross errors and outlying data points are usually tested on artificially generated datasets with variable amount of large outliers generated with different models [1–3, 24]. In this article we intend to investigate the effectiveness of such algorithms on real data without any additional contamination. This is due to the fact that in real problems we do not know whether the data is contaminated, or reliable, when the MLP model is built.

In this work we consider two groups of approaches to deal with this problem. The first group is based on the modification of the neural network parameters, as the error function or neuron input function. The network training is modified and the training data is left in its original state. We discuss these approaches in section 2. The second group of approaches is based on outlier reduction methods. In this case the data is modified and the network is trained using the standard mean square error function. This is discussed in section 3. We also present an algorithm based on joining the aforementioned approaches. Section 4 presents the experimental comparison of the discussed methods on several classification and regression tasks and finally the last section concludes this work.

## 2 Modification of the Network Parameters

### 2.1 Outlier Dependent Error

The simplest approach to make the MLP training process more robust to outliers is to replace common MSE (mean squared error) criterion with a function based on the idea of robust statistical methods. The MSE function is typically used for supervised neural networks training methods because it is simple and easy to optimize error measure. However, similarly to the least mean squares method, it is optimal only for data sets contaminated at most by Gaussian white noise [9, 11, 19]. This is due to the fact that MSE is strongly influenced by large errors. In the case of network training, this influence, measured by a derivative with respect to residuals, can be described by a linear function [9, 17]. To overcome the problem of unpredictable MLP model for training data containing outliers, several robust learning algorithms have been proposed. Such algorithms very often make use of modified error function, derived from robust statistical estimators. The training data are not filtered, so the robustness to outliers is based only on reducing the impact of large training residuals, potentially caused by outlying data points.

The error training function can be modified in many ways: in [17] Liano proposed a new LMLS (Least Mean Log Squares) error function based on so-called M-estimators, which should be optimal for the Cauchy distribution but performs well also for other long-tailed error distributions. Chen and Jain [1] decided to use the Hampel's hyperbolic tangent with additional scale estimator  $\beta$ . The scale estimator helped in determining the range of residuals believed to be outliers. A similar error performance function combined with the annealing



scheme to decrease  $\beta$  with the training progress was proposed by Chuang and Su [2]. A more sophisticated approach, using tau-estimators was described in [20]. Also quartile-based estimators were applied as the error function in the LTS (Least Trimmed Squares) algorithm [23] and in [3], where El-Melegy *et al.* presented the Simulated Annealing for Least Median of Squares (SA-LMedS) algorithm. Similar median error function was described in [24]. The error measure based on robust estimators was also combined with approaches known from image processing, as random sample consensus algorithm [4–6].

All the aforementioned methods focus mainly on modifying the error function, in order to decrease the influence that outliers may have on the network training. In this article we decided to test the most popular robust error measure, namely LMLS. This modification of the training algorithm is a highly cited technique [1–3, 6] and this is why we chose to use it. The LMLS error is then defined as:

$$E_{LMLS}(\mathbf{w}) = \sum_{k=1}^n \sum_{i=1}^m \log(1 + \frac{1}{2} r_{ki}^2(\mathbf{w})), \quad (1)$$

where  $r_{ki} = (y_{ki}(\mathbf{w}) - t_{ki})$  is the error of  $i$ -th output for the  $k$ -th training set element,  $n$  is the size of the training set and  $m$  is the number of network outputs (see fig. 1-left).

Our experiments were performed also with the mean absolute error (MAE) function. This well-known error formula can be also derived from robust M-estimators. As it was demonstrated in [3], the MAE criterion is probably the most effective of all constant error functions, when applied to training data with artificially introduced outliers. We define mean absolute error as:

$$E_{MAE}(\mathbf{w}) = \sum_{k=1}^n \sum_{i=1}^m |r_{ki}(\mathbf{w})|. \quad (2)$$

## 2.2 Median Input Function

An approach to MLP network training using the median neuron input function (MIF) was proposed in [22]. In such networks summation of weighted input signals is replaced with their median. When the summation is replaced by more robust operation, such as median, the neuron output becomes less sensitive to the changes in the input (neuron input, or input weights). Hence, the MIF is not a direct method to make the training process more robust to outliers but it enables the MLP to build a more general model.

Then we can define the MIF neuron output as:

$$y_{out} = f(\text{med}\{w_i x_i\}_{i=1}^N), \quad (3)$$

where  $f(\cdot)$  denotes neuron transfer function (e.g. sigmoid or linear),  $x_j$  are neuron inputs,  $w_i$  is the  $i$ -th input weight and  $N$  denotes input size. However, there exist several problems concerning practical use of MIF networks. It is clearly evident that calculating MIF output is computationally more expensive than in the case

of simple sum. Moreover, the input function given by (3) makes the network error function non-differentiable, so it cannot be simply trained with gradient-based methods. So for that purpose in [22] an approximated algorithm, based on the gradient for a simple sum, was described. This approach can be applied also for non-differentiable error performance based on the median of residuals [24].

Another problem with a MIF is that when the network is trained on regression tasks, the model built by the network may not react to single or small changes in the dataset. For that reason we combine the median input (which provides the outlier-robust part) with the sum (which provides high sensibility), defining a new input function as:

$$y_{out} = f(\delta \text{med}\{w_i x_i\}_{i=1}^N + (1 - \delta) \sum_{i=1}^N w_i x_i), \quad (4)$$

where  $0 < \delta < 1$  determines the median influence on neuron input function. We experimentally determined that the optimal range of  $\delta$  for most datasets is between 0.6 and 0.9. The learning process was not sensible to little changes of  $\delta$  within this range, so in the experiments we used  $\delta = 0.75$ .

### 3 Outlier Reduction

#### 3.1 Instance Selection

The reasons for reducing the number of instances in the training set include: noise reduction by elimination outliers, reducing the data set size and sometimes improving generalization by eliminating instances that are too similar to each other, faster training of the model on a smaller dataset and faster prediction in of the model, especially in the case of lazy-learning algorithms, as k-NN. Thus the aim of the instance selection algorithms can be either noise reduction, as in the case of the Edited Nearest Neighbor algorithm (ENN) [27] or data compression, as in the case of the Condensed Nearest Neighbor rule (CNN) [10] or both. In this work we focus on the first area: noise reduction. Although in practical applications preliminary attribute selection is also beneficial [7] (irrelevant attributes may produce false positive signals in outlier detection algorithms), we do not discuss the aspect here for the sake of focusing on the main topic this work.

A large survey including almost 70 different algorithms of instance selection for classification tasks can be found in [25]. The instance selection issue for regression tasks is much more complex. In classification tasks only the boundaries between classes must be determined, while in regression tasks the output value must be assessed at each point of the input space. Moreover, in classification tasks there are at most several different classes, while in regression tasks, the output of the system is continuous, so there are an unlimited number of possible values to be predicted by the system. The decision about rejection of a given vector in classification tasks can be made based on a right or wrong classification of the vector. In regression problems, rather a threshold defining the difference between the predicted and the actual value should be set. Determining the threshold

(which is rather a function than a constant value) is an important point. Another issue is the measure of the quality of the model, which in classification tasks is very straightforward (classification accuracy), while in regression tasks, it can be defined in several ways. In practical solutions not always the simple error definitions as the MSE (mean square error) work best [14], because also the cost of the error must be taken into account. Depending on a given application, the cost can be higher in different areas of the output (in a similar way as cost matrix in classification tasks) and can depend on the error value. Because of the challenges, there were very few approaches in the literature to instance selection for regression problems. Moreover, the approaches were verified only on artificial datasets generated especially for the purpose of testing the algorithms. Zhang [28] presented a method to select the input vectors while calculating the output with k-NN. Tolvi [26] presented a genetic algorithm to perform feature and instance selection for linear regression models. In their works Guillen et al. [8] discussed the concept of mutual information used for selection of prototypes in regression problems.

---

**Algorithm 1.** regENN algorithm
 

---

**Require:**  $\mathbf{T}$

```

 $m \leftarrow \text{sizeof}(\mathbf{T});$ 
for  $i = 1 \dots m$  do
   $\bar{Y}(\mathbf{x}_i) = \text{NN}((\mathbf{T} \setminus \mathbf{x}_i), \mathbf{x}_i);$ 
   $S \leftarrow \text{Model}(\mathbf{T}, \mathbf{x}_i)$ 
   $\theta = \alpha \cdot \text{std}(Y(\mathbf{X}_S))$ 
  if  $|Y(\mathbf{x}_i) - \bar{Y}(\mathbf{x}_i)| > \theta$  then
     $\mathbf{T} \leftarrow \mathbf{T} \setminus \mathbf{x}_i$ 
  end if
end for
 $\mathbf{P} \leftarrow \mathbf{T}$ 
return  $\mathbf{P}$ 

```

---

For the purpose of noise reduction we will use the ENN (Edited Nearest Neighbor) algorithm [27] and its version for the regression tasks - regENN. The main idea of the ENN algorithm is to remove a given instance if its class is different than the majority class of its neighbors. ENN starts with the whole original training set  $\mathbf{T}$ . Each instance, which is wrongly classified by its  $k$  nearest neighbors is removed from the dataset, as it is supposed to be an outlier. In repeated ENN, the process of ENN is iteratively repeated as long as there are any instances wrongly classified. In all k-NN algorithm, the ENN is repeated for all  $k$  from  $k = 1$  to  $k_{max}$ . In [13] we proposed an extension of several instance selection algorithms for regression tasks. We shortly describe below the ENN algorithm for regression tasks - regENN.

To adjust the ENN algorithms to regression tasks the wrong/correct classification decision is replaced with a distance measure and a similarity threshold, to decide if a given vector can be considered as similar to its neighbors. We use

a weighted k-NN with  $k = 9$ , where the weight  $w_i$  exponentially decreases with the distance  $d_i$  between the given vector and its  $i$ -th neighbor  $x_i$ . The predicted output  $y$  is given by eq. 1.

$$y = \frac{\sum_{i=1}^k w_i y_i}{\sum_{i=1}^k w_i} \quad (5)$$

where  $w_i = 2^{-0.2d_i}$ . We use Euclidean distance measure and a threshold  $\Theta$ , which expresses the maximum difference between the output values of two vectors to consider them similar. Using  $\Theta$  proportional to the standard deviation of  $k$  nearest neighbors of the vector  $x_i$  reflects the speed of changes of the output around  $x_i$  and allows adjusting the threshold to that local landscape, what, as the experiments showed, allows for obtaining higher compression of the dataset. As the regression model to predict the output  $Y(x_i)$  we use k-NN with  $k = 9$  as the Model(T,  $x_i$ ) ( $k = 9$  usually produced good results). In case of regression we experimentally evaluated the optimal  $\Theta$  and we used  $\Theta$  equal to 5 standard deviations of the 9 nearest neighbors for RegENN.

### 3.2 Anomaly Detection

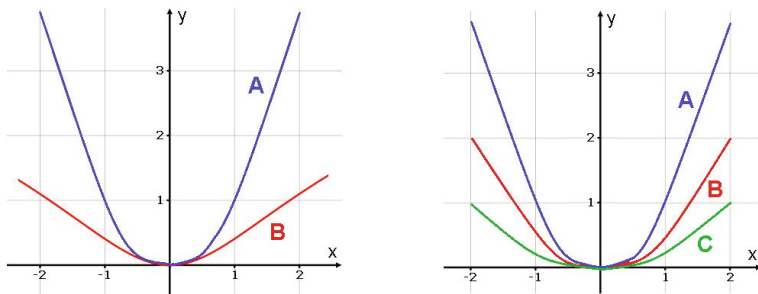
We used the ENN and regENN algorithms to reject the outliers prior to the network training. Here we describe an algorithm based on a k-NN Global Anomaly Score algorithm (k-NN GAS), which we use to assess the degree to which a given instance is an outlier prior to the network training and then remain all the instances in the training set, but differentiate the way they are included in the training. The k-NN Global Anomaly Score algorithm calculates the anomaly score based on the k nearest neighbors implementation. The outlier score of an instance is the average distance between the instance and its k nearest neighbors. In the experiments we use  $k = 9$  and Euclidean distance measure. The higher the outlier score the more anomalous the instance is. However, for the purpose of labeled data, we had to extend this score, including for the calculation the distance in the input space  $d_x$  and the distance in the output space  $d_y$ . In the case of classification we add one to  $d_y$  for each neighbor of a different class and zero for each neighbor of the same class. We define the modified anomaly score  $A_{sc}$  as:

$$A_{sc} = d_y/d_x \quad (6)$$

Then we assume that the higher the anomaly score is, the more likely the instance is to be an outlier and the less influence it should have on the network training. We obtain this by dividing the error the network makes during the training on each instance by a greater value if the instance anomaly score for the instance is higher (see fig. 1-right):

$$Error = \begin{cases} Error/A_{sc}^2 & \text{if } A_{sc} > \alpha \text{median}(A_{sc}) \\ Error/(\alpha \text{median}(A_{sc})) & \text{otherwise} \end{cases} \quad (7)$$

Where  $\alpha$  is a parameter. In the experiments we used  $\alpha = 1$ . This modification (considering also distance in the output space) of the k-NN global anomaly score incorporates the idea of the local density of Local Outlier Probability detection algorithm (LOOP). LOOP, contrary to k-NN global anomaly score, includes only the local density of the points in the hyperspace. If a density measured by the number of points in a hypersphere of a certain radius, where the given point is in the center of the hypersphere, is much smaller than inside hyperspheres centered upon k-neighbors of that point, then the point is considered an outlier. The resulting values are scaled to a value range of (0;1). The higher the value the more anomalous the instance is. A survey of outlier detection methods can be found in [12].



**Fig. 1.** Left: The square (A) and the LMLS (B) error function. Right: The square error function used for non-outliers (A), for a weak outlier (B) and for a strong outlier (C).

## 4 Experimental Comparison

### 4.1 Datasets

It is worth noticing that all the considered datasets were not artificially contaminated. We did not introduce artificial outliers, testing the algorithms on the real data. We performed the experiments on eight datasets. First all the datasets were standardized so that the mean value of each attribute is zero and the standard deviation is one to make comparison of the results easier. We used four classification and four regression datasets. Six datasets come from the UCI Machine Learning Repository [18]: Iris (3 classes, 4 attributes, 150 instances), Diabetes (2 classes, attr., inst.), Glass (5 classes, attr., inst.), Ionosphere (2 classes, attr., inst.), Concrete Compression Strength (regression, 7 attributes, 1030 instances), Crime and Communities (regression, 7 attr., 320 inst.). Two datasets comes from a metallurgical industry. The purpose of the SteelC dataset (regression, 14 attr., 2384 inst.) is to predict the amount of carbon that must be added in the steel-making process, given various chemical and physical properties of the liquid steel in the furnace. The purpose of the SteelT dataset (regression, 11 attr., 7401 inst.) is to predict the actual temperature of the liquid steel given a set of physical

values, as temperature in various points on the surface of the furnace, the energy delivered to the process and others (directly measuring the temperature requires some disruptions of the steel-making process making it longer and more expensive).

## 4.2 Experimental Setup

We implemented the algorithms in C# and Matlab. The source code and datasets used in the experiments can be downloaded from [16]. The whole process in different configurations was run in 10-fold crossvalidation loops. To be able to compare the results, we always measure and report in the table the MSE error on the test sets, no matter which error function was used for the network training. Also the MLP architecture was constant (the same for each training method) for a given dataset. We used 5 hidden neurons for iris, diabetes, glass, steelC, and crime, and 6 hidden neurons for the ionosphere, steelT and concrete.

## 4.3 Modification of the Network Parameters with Gradient-Based Learning

In our experiments we decided to apply two general training frameworks: gradient-based learning for modified network parameters (different error and neuron input functions), and non-gradient VSS method for the rest of the algorithms. Tested error functions included LMLS and MAE error measures (equations 1 and 2), used as MLP training criteria. We used also MIF (3), and MedSum (4) neuron inputs. For the case of MedSum, we assumed equal participation of both input functions, setting  $\delta = 0.5$ . The results of such modified networks were compared to traditional approach: the MSE error function and simple weighted sum as neuron input.

For the modified error functions we could not use one of the most popular methods, namely, Levenberg-Marquardt algorithm, which is dedicated to the MSE. Moreover, as it was mentioned in Section 2.2, the MIF nets cannot be trained by regular gradient methods, so following [24, 22] we decided to apply resilient backpropagation (Rprop) algorithm [21].

Analyzing the results obtained for these methods, we cannot definitively decide that there is a single method, which outperforms other approaches. The standard MSE network is the best between gradient-based methods only for one dataset (steelC), similarly MIF (ionosphere), whereas MAE, LMLS, and MedSum are the winners for two datasets each. (As mentioned earlier we compared the MSE on the test sets, no matter which error function was used for network training.) The performances of the tested approaches differ, depending on the training data sets. Only pure MIF strategy, because of its discontinuity, often worsen the results. In general, these algorithms seem to be more efficient, than outlier reduction methods, for classification tasks, and less accurate for regression problems.

#### 4.4 Outlier Reduction with Nongradient-Based Learning

The training algorithm we used to test the outlier reduction-based method was a non-gradient based VSS (Variable Step Search Algorithm). The idea of the VSS algorithm is to make advantage of the properties of MLP network error surface in the weight space, on which the training trajectory is situated, that it is more likely that each training epoch the trajectory direction will be only slightly adjusted that totally changed and therefore the VSS algorithm makes guesses about the optimal modification of each neuron weights in each epoch and then the guesses are adjusted as needed. Details of the algorithm can be found in [15]. The VSS algorithm was also used to test the coexistence of outlier reduction methods and the MedSum function. We used the same network architectures as for the Rprop training and about 12 training epochs for each dataset. We are going to join two our programs in one and use only one training method in the future experiments. However, to obtain a clear comparison between the methods tested with Rprop and with VSS, the exact number of epochs was adjusted so that the crossvalidation accuracy on the standard network trained with Rprop and VSS would be the same.

Looking at the results obtained for these methods, we can observe some regularity. First, the ENN usually did not improve the results with the classification problems, even more the results were worse. This can be explained by the fact,

**Table 1.** Experimental results for classification problems - classification accuracy in 10-fold crossvalidation (higher is better)

algorithm	iris	diabetes	glass	ionosphere
MSE	0.963±0.038	0.765±0.046	0.670±0.094	0.911±0.051
LMLS	0.971±0.055	0.773±0.044	0.691±0.089	0.899±0.083
MAE	0.951±0.063	<b>0.784±0.052</b>	0.592±0.082	0.828±0.061
MIF	0.940±0.095	0.695±0.061	0.605±0.102	<b>0.950±0.048</b>
MedSum	0.956±0.058	0.760±0.037	<b>0.696±0.104</b>	0.922±0.060
ENN	0.960±0.040	0.765±0.043	0.614±0.066	0.910±0.059
k-NN GAS	<b>0.974±0.045</b>	0.775±0.031	0.674±0.074	0.911±0.029
MedSum + k-NN GAS	0.967±0.032	0.764±0.047	0.684±0.046	0.917±0.053

**Table 2.** Experimental results for regression problems - MSE in 10-fold crossvalidation (lower is better)

algorithm	steelC	steelT	concrete	crime
MSE	0.031±0.013	0.576±0.111	0.798±0.250	0.343±0.062
LMLS	0.033±0.019	0.551±0.072	0.857±0.296	0.345±0.101
MAE	0.038±0.013	0.560±0.082	0.969±0.358	<b>0.318±0.109</b>
MIF	0.070±0.018	0.844±0.070	0.869±0.267	0.341±0.060
MedSum	0.035±0.015	0.574±0.113	0.794±0.321	0.338±0.080
regENN	0.032±0.010	0.576±0.111	0.778±0.256	0.341±0.067
k-NN GAS	0.031±0.018	0.541±0.097	0.778±0.251	0.341±0.060
MedSum + k-NN GAS	<b>0.028±0.013</b>	<b>0.526±0.092</b>	<b>0.764±0.211</b>	0.336±0.073

that the instances, which were rejected by ENN were in most cases situated close to the decision boundaries and thus removing them made determining the class boundaries less precise. It would probably work better with very noisy data, where it would reject more outliers that vectors situated close to class boundaries. In case of regression the regENN algorithm can be adjusted with the  $\Theta$  parameter to reject the optimal number of outliers (if the error increases we increase  $\Theta$ , till the error stop increasing), so it does not cause error increase. However it decreased the error only in two out of four cases. The modified k-NN GAS algorithm adjusts the error measure individually to each training vector. Because it can be adjusted with the  $\alpha$  parameter it also did not worsen the results, but it improved them in all but two cases. Joining the modified k-NN GAS algorithm with the MedSum neuron input function proved to work best. The results were further improved in six out of eight cases. This can be explained by the fact that when to the summation in the neuron function a more stable operation, such as median is added, the neuron output becomes less sensitive to perturbances in the data.

## 5 Conclusions

We examined several methods of improving the performance of MLP neural networks by outlier elimination and compare them experimentally on several classification and regression tasks. The examined method included: pre-training outlier elimination, use of different error measures during network training, replacing the weighed input sum with weighed median in the neuron input functions and various combinations of them.

The obtained results are very interesting. There is an additional cost of calculating the median equal to the quicksort algorithm of sorting the arrays of the dimension of the number of instances in the training set. Especially joining the two groups of methods in one network training, allowed for the best error reduction, especially in regression problems. In most cases, modified MLP training methods outperformed traditional MSE approach, however the choice of the best algorithm can be merely indicative. Especially for the regression problems most of the best results were obtained for our novel hybrid algorithm, joining k-NN GAS with MedSum neuron input. This approach can be considered as effective tool to improve the MLP performance on real-life problems.

## References

1. Chen, D., Jain, R.: A robust backpropagation learning algorithm for function approximation. *IEEE Transactions on Neural Networks* 5(3), 467–479 (1994)
2. Chuang, C.C., Su, S.F., Hsiao, C.C.: The annealing robust backpropagation (arbp) learning algorithm. *IEEE Transactions on Neural Networks* 11(5), 1067–1077 (2000)
3. El-Melegy, M.T., Essai, M.H., Ali, A.A.: Robust training of artificial feedforward neural networks. In: Hassaniien, A.-E., Abraham, A., Vasilakos, A.V., Pedrycz, W. (eds.) *Foundations of Computational, Intelligence Volume 1. SCI*, vol. 201, pp. 217–242. Springer, Heidelberg (2009), [http://dx.doi.org/10.1007/978-3-642-01082-8\\_9](http://dx.doi.org/10.1007/978-3-642-01082-8_9)



4. El-Melegy, M.: Random sampler m-estimator algorithm for robust function approximation via feed-forward neural networks. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 3134–3140 (2011)
5. El-Melegy, M.: Ransac algorithm with sequential probability ratio test for robust training of feed-forward neural networks. In: In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 3256–3263 (2011)
6. El-Melegy, M.: Random sampler m-estimator algorithm with sequential probability ratio test for robust function approximation via feed-forward neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 24(7), 1074–1085 (2013)
7. Golak, S., Burchart-Korol, D., Czaplicka-Kolarz, K., Wiczorek, T.: Application of neural network for the prediction of eco-efficiency. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) *ISNN 2011, Part III. LNCS*, vol. 6677, pp. 380–387. Springer, Heidelberg (2011), [http://dx.doi.org/10.1007/978-3-642-21111-9\\_43](http://dx.doi.org/10.1007/978-3-642-21111-9_43)
8. Guillen, A.: Applying mutual information for prototype or instance selection in regression problems. In: *ESANN 2009* (2009)
9. Hampel, F.R., Ronchetti, E.M., Rousseeuw, P.J., Stahel, W.A.: *Robust Statistics: The Approach Based on Influence Functions* (Wiley Series in Probability and Statistics), revised edn. Wiley Interscience, New York (April 2005), <http://www.worldcat.org/isbn/0471735779>
10. Hart, P.: The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory* 14(3), 515–516 (1968)
11. Huber, P.J.: *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley-Interscience (1981), <http://www.worldcat.org/isbn/0471418056>
12. Ben-Gal, I.: *Outlier detection*. Kluwer Academic Publishers (2005)
13. Kordos, M., Białka, S., Blachnik, M.: Instance Selection in Logical Rule Extraction for Regression Problems. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) *ICAISC 2013, Part II. LNCS*, vol. 7895, pp. 167–175. Springer, Heidelberg (2013), [http://dx.doi.org/10.1007/978-3-642-38610-7\\_16](http://dx.doi.org/10.1007/978-3-642-38610-7_16)
14. Kordos, M., Blachnik, M., Wiczorek, T.: Temperature prediction in electric arc furnace with neural network tree. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) *ICANN 2011, Part II. LNCS*, vol. 6792, pp. 71–78. Springer, Heidelberg (2011), [http://dx.doi.org/10.1007/978-3-642-21738-8\\_10](http://dx.doi.org/10.1007/978-3-642-21738-8_10)
15. Kordos, M., Duch, W.: Variable step search algorithm for feedforward networks. *Neurocomputing* 71(13–15), 2470–2480 (2008), <http://www.sciencedirect.com/science/article/pii/S0925231208002099>
16. Kordos, M.: source code and datasets used in the experiments (2013), <http://www.ath.bielsko.pl/~mkordos/tpnc2013.html>
17. Liano, K.: Robust error measure for supervised neural network learning with outliers. *IEEE Transactions on Neural Networks* 7(1), 246–250 (1996)
18. Merz, C., Murphy, P.: *Uci repository of machine learning databases* (2013), <http://www.ics.uci.edu/mllearn/MLRepository.html>
19. Olive, D.J., Hawkins, D.M.: *Robustifying robust estimators* (2007)
20. Perní-Espinoza, A.V., Ordieres-Meré, J.B., de Poisoń, F.J.M., González-Marcos, A.: Tao-robust backpropagation learning algorithm. *Neural Networks* 18(2), 191–204 (2005), <http://www.sciencedirect.com/science/article/pii/S0893608004002345>

21. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: the rprop algorithm. *IEEE International Conference on Neural Networks* 1, 586–591 (1993)
22. Rusiecki, A.: Fault tolerant feedforward neural network with median neuron input function. *Electronics Letters* 41(10), 603–605 (2005)
23. Rusiecki, A.: Robust LTS backpropagation learning algorithm. In: Sandoval, F., Prieto, A.G., Cabestany, J., Graña, M. (eds.) *IWANN 2007*. LNCS, vol. 4507, pp. 102–109. Springer, Heidelberg (2007), [http://dx.doi.org/10.1007/978-3-540-73007-1\\_13](http://dx.doi.org/10.1007/978-3-540-73007-1_13)
24. Rusiecki, A.: Robust learning algorithm based on iterative least median of squares. *Neural Processing Letters* 36(2), 145–160 (2012), <http://dx.doi.org/10.1007/s11063-012-9227-z>
25. Salvador, G., Derrac, J., Ramon, C.: Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 417–435 (2012)
26. Tolvi, J.: Genetic algorithms for outlier detection and variable selection in linear regression models. *Soft Computing* 8, 527–533 (2004)
27. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on SMC Systems, Man and Cybernetics* 2(3), 408–421 (1972)
28. Zhang, J.: Intelligent selection of instances for prediction functions in lazy learning algorithms. *Artificial Intelligence Review* 11, 175–191 (1997)

# A Trajectory Algorithm to Solve the Relay Node Placement Problem in Wireless Sensor Networks

Jose M. Lanza-Gutiérrez, Juan A. Gomez-Pulido,  
and Miguel A. Vega-Rodríguez

Department of Computers and Communications Technologies,  
University of Extremadura, Polytechnic School, Campus Universitario s/n, 10003  
Cáceres, Spain  
{jmlanza,jangomez,mavega}@unex.es

**Abstract.** Nowadays, wireless sensor networks are widely used in many fields of application. This promotes that many authors try to overcome the most important shortcomings of this type of network. This paper focuses on how to add relay nodes to previously established static wireless sensor networks in order to optimize two important factors: average energy consumption and average coverage. Since this is an NP-hard optimization problem, three different multiobjective metaheuristics are used; two of them are well-known genetic algorithms (NSGA-II and SPEA2) and the third is a multiobjective version of the trajectory algorithm VNS. All the results obtained are analyzed by means of a widespread statistical methodology, using both set coverage and hypervolume as multiobjective quality metrics. We conclude that MO-VNS provides better performance on average than standards NSGA-II and SPEA2.

**Keywords:** metaheuristics, optimization, relay node, sensors.

## 1 Introduction

At present, Wireless Sensor Networks (WSNs) are one of the most important emerging wireless technologies. We can find examples of its use in a wide range of fields, such as environmental control, fire detection, precision agriculture, robotic, home automation, rescue operations, among others [13].

Traditionally, WSNs are composed of a set of sensors that capture information (temperature, humidity, ...) of its environment, and a sink node (also called collector node) that collects all the captured information, being this node the only connection point of the WSN to the outside. There are some important features that encourage the use of WSNs, such as the absence of wires and the use of power autonomous low cost devices. These factors, among others, allow us to deploy WSNs in environments where the use of other networks would be very expensive or impossible. Nevertheless, WSNs also have important shortcomings which have not been solved yet and affect important features.

WSNs have a clear dependence on the energy consumption because sensors are powered by batteries to avoid the use of wires: each time a sensor captures an information packet, this device has to send it to the collector node, which

implies an energy expenditure. In addition, sensors are usually at a distance greater than one hop from the collector node, which involves a higher energy cost due to retransmissions, and causes the existence of bottlenecks (sensors that are subjected to a higher energy cost). With the purpose of avoiding these bottlenecks, a new device specialized in communication tasks and called router or relay node was recently added to WSNs, reducing the workload of sensors.

The efficient deployment of a WSN depends on many factors, even more if we include these new devices. For this reason, the efficient design of a WSN has been defined as an NP-hard optimization problem by some authors [2] [16]. NP-hard optimization problems have been tackled in the literature through several methodologies, e.g heuristics, Evolutionary Computation (EC), and so on.

We find two main lines of research for WSNs. On the one hand works that study the optimization of traditional WSNs. Using heuristics we may cite the contributions of Cheng et al.[2](who optimize the energy consumption assigning different power transmission levels to the sensors) and Cardei et al [1] (who split WSNs into disjoint set of sensors, deciding which must be active at each time to increase network lifetime); and using EC Konstantinidis and Yang [6] (who assign different power transmission levels to the sensors as in [1], but optimizing coverage and network lifetime). On the other hand, works that optimize WSNs by adding relay nodes. This line of research appeared later trying to overcome an important shortcoming: most papers in the previous line use many redundant sensors to optimize the energy consumption, which implies a higher network cost. Using heuristics, we can find papers such as Han et al. [5](who maximize the network lifetime) and Wang et al. [16] (who optimize the energy consumption using routers with processing limitations); and using EC Perez et al. [14](who optimize the energy cost and the number of routers used.)

Our work follows the second line of research: we study how to add routers to previously-established traditional static WSNs (a set of sensors and a collector node) simultaneously optimizing two important factors: average energy consumption and average coverage; this is the so-called Relay Node Placement Problem (RNPP). With this goal, we used three different metaheuristics (techniques to solve very general kind of optimization problems), providing good solutions in lesser computation times.

In the course of this paper, the following contributions are presented:

- The RNPP is solved using a multiobjective version of the Variable Neighborhood Search algorithm (MO-VNS) [4], comparing the results obtained with a previous contribution [9] where two well-known standard genetic algorithms were used: NSGA-II [3] and SPEA-2 [19].
- We used two multiobjective measures (hypervolume[17] and set coverage[18]), and a widespread statistical methodology to analyze the results obtained.
- A public free testbed was used [8], which means that this work can be replicated and improved by other authors. In the literature, most papers use randomly generated data set or non-public ones.

The organization of the rest of the paper is as follows. In Section 2 we give a formal statement of the RNPP. Algorithms used are described in Section 3 and

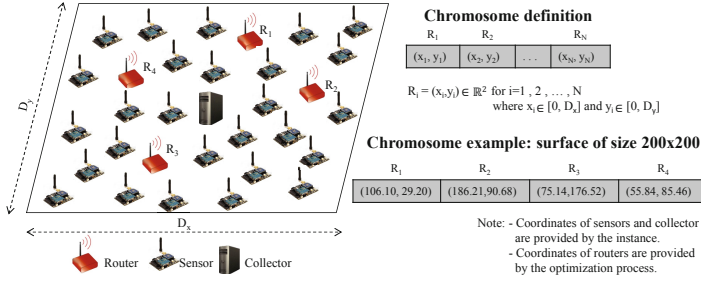


Fig. 1. Representation of the network model considered in this work

the simulation results in Section 4. Finally, we expose our conclusions and future works in Section 5.

## 2 Relay Node Placement Problem in WSNs

A WSN as we consider in this paper is composed of a 2D-surface of size  $D_x \times D_y$  where three kinds of devices are placed:  $M$  sensors,  $N$  routers and a sink node (see Figure 1). Sensors capture information from its environment with a sensibility radius  $R_s$ , on regular basis and simultaneously. Each time a sensor capture an information packet (with  $K$  bytes), this device must send it to the sink node which collects all the information captured, using as routing protocol the minimum distance among devices provided by Dijkstra’s algorithm. Finally, routers are devices specialized in communication tasks, which only relay all received information to the collector node. All these devices can communicate among them with a same communication radius  $R_c$ , having collector and routers an unlimited power supply, and sensors being powered by batteries.

With the purpose of modeling the energy expenditure suffered by the sensors, a known energy model proposed by A. Konstantinidis et al. was used [6]. This model simulates packet loss, which is common in wireless environments. In addition, we suppose a perfect medium access and a perfect synchronization among devices, which ensures that there will be no collisions among communications.

According to this energy model, the transmission power that a sensor  $i$  needs to reach another device  $j$  is defined as

$$P_i = \beta \cdot d_{i,j}^\alpha \quad i \in 1, \dots, M \quad j \in 1, \dots, M + N + 1, \quad (1)$$

where  $\beta > 0$  is the transmission quality parameter,  $d_{i,j}$  is the distance between  $i$  and  $j$ , and  $\alpha \in [2, 4]$  is the path loss exponent. All the sensors start with the same energy charge ( $EIC$ ) in their batteries, which will be decreased over time due to transmission tasks. Following this definition, the residual energy of a sensor  $i$  at time  $t$  is given by the following expression

$$E_i(t) = E_i(t - 1) - [(r_i(t) + 1) \cdot P_i \cdot amp * K] \quad i \in 1, \dots, M, \quad (2)$$

where  $amp$  is the energy consumption per bit of the power amplifier,  $r_i(t)$  is the number of incoming packets that  $i$  receives at time  $t$ , and that the sensor has to relay to the collector node; and the  $+1$  term is due to  $i$  captures an information packet at this time which must be sent too. As other works consider, the energy cost due to capturing, processing and reception tasks are considered negligible.

There is an important concept in this problem definition: the network lifetime ( $LF$ ). It is the amount of time units over which a WSN is able to provide enough information about its environment. A coverage threshold ( $CV$ ) is used for this purpose, i.e at least a certain percentage of the surface must be covered to consider the information provided is enough. When a sensor battery is exhaust, the coverage provided by the sensor will not be taking into account, and then the global coverage could decrease reaching the coverage threshold.

To summarize, the RNPP is defined as an NP-hard multiobjective optimization problem given by the size of the 2D-surface ( $D_x \times D_y$ ), the number of routers ( $N$ ), the position of the collector node ( $C$ ), the position of the  $M$  sensors, the information packet size ( $K$ ), the sensibility and communication radius ( $R_s$  and  $R_c$ ), the coverage threshold ( $CV$ ), the initial energy charge ( $IEC$ ) and the energy parameters ( $amp$ ,  $\beta$  and  $\alpha$ ). The fitness functions are the following:

- Average Energy Consumption (AEC): This is the average power consumption of the sensors over the network lifetime. It is measured in Joules (J). This function is defined as

$$f_1 = LF^{-1} \times \sum_{t=1}^{LF} \sum_{i=1}^M \left( \frac{E_i(t-1) - E_i(t)}{sensorsAlive(t)} \right), \quad (3)$$

where  $sensorsAlive(t)$  is the number of sensors with a residual energy greater than zero at time  $t$ , and  $E_i(t)$  is given by (2).

- Average Coverage (EC): This is the average coverage provided by the sensors over the network lifetime. In the literature, we can find two main ways to obtain the network coverage. On the one hand, some authors say that a sensor covers a round area of radius  $R_s$ , so the global coverage is the union of each of them. On the other hand, other authors use a binary matrix of demand points placed on the surface, where a demand point will be active whether there is some alive sensor at a distance lower than  $R_s$  at this time; finally, activated demand points are counted. In this work, we opted for the second method; although the first one is a bit more precise, the second one is less hard to compute. The fitness function is defined as

$$f_2 = LF^{-1} \times \sum_{t=1}^{LF} \sum_{x=1}^{\lceil D_x \rceil} \sum_{y=1}^{\lceil D_y \rceil} \left( \frac{R_{x,y}(t)}{\lceil D_x \rceil \times \lceil D_y \rceil} \right), \quad (4)$$

where  $R$  is a matrix of  $\lceil D_x \rceil \times \lceil D_y \rceil$  cells with a binary demand point placed in the middle of each cell. Accordingly,  $R_{x,y}(t)$  is the demand point placed in the cell  $\{x,y\}$  at time  $t$ , so the cell size is  $1 \times 1$ . A value of zero implies that the demand point is inactive, and a value of one is the opposite.

### 3 Multiobjective Optimization

As we said previously, the RNPP is an NP-hard problem, hence it is necessary to use non-traditional techniques to solve it in a reasonable computing time. With this purpose, we decided to use three different metaheuristics, two of them (NSGA-II and SPEA-2) are genetic algorithms, and the third is a multiobjective version of the trajectory algorithm VNS. This last algorithm explores the search space by moving from one neighborhood to another, showing a good performance in many optimization problems. In this paper, a MO-VNS version which includes a perturbation mechanism is used, because of we checked that it provided better results for this problem definition than the traditional version without this mechanism. These results are not included due to space limitations.

NSGA-II and SPEA-2 follow a similar scheme. NSGA-II uses two populations  $P_t$  and  $Q_t$  of equal size PS; the first one saves the parents of generation  $t$ , and the second one saves the offspring of population  $P_t$ . Initially,  $P_t$  is randomly generated and  $Q_t$  is empty, then so long as the stop criterion is not reached, both populations are combined in a new one  $R_t$ . Next all solutions in  $R_t$  are evaluated using crowding and rank measures, selecting the best PS solutions of  $R_t$  as the new  $P_{t+1}$ . Then a new  $Q_{t+1}$  is generated based on  $P_{t+1}$ . For this purpose, and so long as the new  $Q_{t+1}$  is not filled, a pair of individuals are selected from  $P_{t+1}$  though binary tournament method. Then, using both crossover and mutation operators, a new individual is generated and inserted into  $Q_{t+1}$ . We considered as crossover operator the widely used one-point crossover. As mutation operator, we used a greedy strategy, where the router coordinates are randomly changed and only changes that provide better solutions are considered.

SPEA-2 uses an auxiliary population  $\overline{P}_t$  where the best solutions are saved along generations, and a regular population  $P_t$  with sizes  $\overline{PS}$  and PS respectively. Initially,  $P_t$  is randomly generated and  $\overline{P}_t$  is empty, then so long as the stop condition is not reached, the algorithm obtains the fitness values for each solution in  $P_t \cup \overline{P}_t$ . This fitness value is based on the dominance concept and an additional density measure, so the algorithm calculates the number of individuals that each individual dominates. Next, according to these fitness values, the best solutions of  $P_t \cup \overline{P}_t$  are inserted into the new  $\overline{P}_{t+1}$ . Finally, a new  $P_{t+1}$  is generated through  $\overline{P}_{t+1}$  using crossover and mutation as we discussed for NSGA-II.

MO-VNS is a multiobjective trajectory algorithm that uses neighborhood structures to perform local searches. The definition of these neighborhood operators is an essential aspect in the algorithm. In this paper, a neighborhood structure is defined as the maximum value that a router can be displaced from its initial coordinates during the local search. Accordingly, the set of neighborhood structures  $N_s$  is given by the expression

$$N_s = \left\{ n_{s_k} \in \mathbb{R} / n_{s_k} = \frac{\min(D_x, D_y) * k}{dv * k_{max}} \right\} \quad \text{for } k = 1, \dots, k_{max}, \quad (5)$$

where  $k_{max}$  is the maximum number of neighborhood structures,  $dv$  is the reduction factor of the surface to delimit the maximum displaced, and  $\min(D_x, D_y)$

---

**Algorithm 1.** MO-VNS with perturbation
 

---

```

1: insert a random solution in the empty Pareto Front  $P_v$ 
2: generate the set of neighborhood structures  $N_s$ 
3: while not stop condition do
4:   while there are some unmarked solution in  $P_v$  do
5:      $a \leftarrow$  randomly pick an unmarked solution from  $P_v$  and mark  $a$ 
6:      $n_{s_k} \leftarrow$  randomly pick a neighborhood structure,  $k \in 1, \dots, k_{max}$ ,  $n_{s_k} \in N_s$ 
7:     while  $k \leq k_{max}$  do
8:        $\tilde{a} \leftarrow$  generate a neighborhood solution of  $a$  in  $n_{s_k}$ 
9:       insert  $\tilde{a}$  in  $P_v$  and update the Pareto Front
10:      if  $\tilde{a} \in P_v$  then
11:         $k \leftarrow 0$  and  $a \leftarrow \tilde{a}$ 
12:      else
13:         $k \leftarrow k + 1$ 
14:      end if
15:    end while
16:  end while
17:  perform perturbation in  $P_v$  to generate a new base solution
18:  reset all the marks of  $P_v$ 
19: end while

```

---

is the minimum value between  $D_x$  and  $D_y$ . Note that  $N_s$  is a sorted set, hence  $n_{s_k}$  will be lower than  $n_{s_{k+1}}$ .

As we observe in Algorithm 1, MO-VNS starts with the generation of an initial solution which is added to the non-dominated solution set  $P_v$  that the algorithm maintains during search (line 1). Randomly, an unmarked solution  $a \in P_v$  and a neighborhood structure  $n_{s_k} \in N_s$  are selected to generate a neighboring solution  $\tilde{a}$  through a local search using  $a \in P_v$  as base solution, marking  $a$  as visited (lines 5-6). This way, for each router  $R_{a_z} \in \mathbb{R}^2$ ,  $z \in 1, \dots, N$  of  $a \in P_v$ , we perform a random displaced defined by the following expression:

$$R_{\tilde{a}_z} = R_{a_z} + \left( \frac{n_{s_k}}{2} - rand(n_{s_k}) \right) \quad n_{s_k} \in N_s, k \in 1, \dots, k_{max}, \quad (6)$$

where  $R_{\tilde{a}_z}$  is the  $z$ -th router of the new individual  $\tilde{a}$  and  $rand(n_{s_k})$  is a random number between 0 and  $n_{s_k}$ . Next,  $\tilde{a}$  is inserted into  $P_v$ , updating  $P_v$  so that  $P_v$  only contains non-dominated solutions (lines 8-9). If  $\tilde{a} \in P_v$ , then the local search provided a good solution, so  $\tilde{a}$  is chosen as base solution and the search is repeated again using a value  $k = 0$  (lines 10-11). Otherwise, the neighborhood structure is increased so long as  $k$  reaches the maximum number of structures  $k_{max}$  (lines 12-13). If all the non-dominated solutions in the current  $P_v$  were explored, the marks will be reset, all individuals are eligible for a reselection, and a perturbation mechanism is performed trying to avoid local minima (lines 17-18). This mechanism is as follows: for each solution in  $P_v$ , a greedy mutation operator is used as previously discussed for NSGA-II and SPEA2.



**Table 1.** Instances used in this paper

Instance	$D_x \times D_y$	M	$R_c$	HO-AEC	HO-AC	Ref AEC		Ref AC	
						ideal	nadir	ideal	nadir
100x100_15_30	100x100	15	30	0.1091	89.24%	0.02	0.1	100%	60%
100x100_15_60	100x100	15	60	0.1482	86.63%	0.02	0.1	100%	60%
200x200_15_30	200x200	57	30	0.2791	87.10%	0.10	0.30	100%	60%
200x200_15_60	200x200	57	60	0.3871	82.43%	0.10	0.30	100%	60%
300x300_15_30	300x300	128	30	0.4225	76.44%	0.04	0.50	100%	60%

**Table 2.** Parametric sweep

NSGA-II		
Parameter	Value	Range
Mutation	0.5	0.05,0.1,0.15,...,0.95
Crossover	0.5	0.05,0.1,0.15,...,0.95
SPEA2		
Parameter	Value	Range
Mutation	0.5	0.05,0.1,0.15,...,0.95
Crossover	0.5	0.05,0.1,0.15,...,0.95
MO-VNS		
Parameter	Value	Range
Mutation	0.1	0.05,0.1,0.15,...,0.95
$k_{max}$	11	4,5,6,...,14
dv	3	1,2,3,4,5

### 4 Simulation Results

With the aim of comparing among the metaheuristics discussed previously, some data sets were used. As none was found that fit this problem definition, a synthetic data set was defined by ourselves in [8]. This data set is composed of three scenarios where a set of sensors and a collector node are placed (Table 1). In all cases, the number of sensors used is the minimum value to cover all the surface, and the collector is placed in the center of the scenario.

As we stated in Section 2, the RNPP is defined by several parameters. In this work, we take:  $R_s = 15m$  and  $R_c = 15m$  from [12],  $R_c = 30m$  (devices with higher communication capacities),  $K = 128KB$ ,  $CV = 70\%$ , and the energy parameters  $\beta = 1$ ,  $EIC = 5J$ ,  $amp = 100pJ/bit/m^2$  and  $\alpha = 2$  from [7].

The algorithms were used to optimize these instances adding routers. As we see in Table 3, 16 test cases were defined, where a test case is an instance in which a fixed number of routers are added. These test cases were defined taking into account that including routers increases network cost; hence we decided not to include more than 20% of these devices with respect to the number of sensors.

Before starting the experiments, the algorithms were tuned by means of a parametric sweep [9]. In Table 2, we show the configurations that provide the best behavior on average for each algorithm, and the range of values considered. The population size for NSGA-II and SPEA2 was an habitual value of 100 individuals. Chromosome definition is shown in Fig. 1.

As the RNPP is a multiobjective problem, we used some multiobjective metrics. On the one hand, the hypervolume was considered. Accordingly, in Table 3 we find both average hypervolume and standard deviation for each algorithm, test case and stop condition, where higher hypervolumes for 400 000 evaluations are highlighted. Hypervolumes were calculated using the reference points  $RefAEC$  and  $RefAC$  listed in Table 1. We performed 31 independent runs for each combination in order to get statistical conclusions, and at least 30 samples are necessary. We analyzed the results by means of a widely used statistical methodology. Firstly, Shapiro–Wilk’s [15] and Kolmogrov–Smirnov–Lilliefors’s [10] tests were used with the hypotheses: data follow a normal distribution ( $H_0$ ) or not ( $H_1$ ). P-values lower than 0.05 were obtained for all the cases, so data do not follow a normal distribution, therefore we must use the median as average

**Table 3.** Hypervolume and standard deviation for each algorithm and test case

MO-VNS ( $\overline{Hyp}$ %, <i>std.dev</i> )						
Test case	Evaluations (Stop condition)					
Instance (routers)	50 000	100 000	200 000	300 000	400 000	
100x100_15_30(2)	41.90%, 0.0279	42.73%, 0.0214	44.68%, 0.0002	44.69%, 0.0001	<b>44.69%</b> , 0.0000	
100x100_15_30(3)	58.28%, 0.0037	58.47%, 0.0006	58.49%, 0.0006	58.52%, 0.0020	<b>58.57%</b> , 0.0016	
100x100_15_60(2)	34.46%, 0.0012	34.56%, 0.0002	34.56%, 0.0001	34.57%, 0.0000	<b>34.57%</b> , 0.0000	
100x100_15_60(3)	60.74%, 0.0148	61.29%, 0.0104	61.62%, 0.0090	62.28%, 0.0010	<b>62.32%</b> , 0.0008	
200x200_15_30(2)	35.44%, 0.0394	37.35%, 0.0325	38.35%, 0.0340	38.86%, 0.0322	<b>41.01%</b> , 0.0010	
200x200_15_30(4)	48.64%, 0.0488	50.13%, 0.0445	51.56%, 0.0368	53.61%, 0.0192	<b>54.69%</b> , 0.0114	
200x200_15_30(6)	66.48%, 0.0132	67.06%, 0.0130	67.47%, 0.0124	67.70%, 0.0120	<b>67.80%</b> , 0.0119	
200x200_15_30(9)	77.99%, 0.0153	78.97%, 0.0160	79.63%, 0.0144	80.16%, 0.0125	<b>80.31%</b> , 0.0145	
200x200_15_60(2)	23.43%, 0.0137	24.26%, 0.0063	24.51%, 0.0045	24.80%, 0.0016	<b>24.88%</b> , 0.0004	
200x200_15_60(4)	61.83%, 0.0044	61.95%, 0.0049	62.16%, 0.0046	62.27%, 0.0044	<b>62.39%</b> , 0.0044	
200x200_15_60(6)	77.06%, 0.0067	77.66%, 0.0056	77.84%, 0.0098	78.06%, 0.0076	<b>78.23%</b> , 0.0074	
200x200_15_60(9)	89.93%, 0.0060	90.46%, 0.0103	91.08%, 0.0096	91.37%, 0.0094	<b>91.43%</b> , 0.0095	
300x300_15_30(6)	41.09%, 0.0149	41.66%, 0.0156	42.18%, 0.0162	42.42%, 0.0151	<b>42.56%</b> , 0.0151	
300x300_15_30(12)	47.31%, 0.0082	47.95%, 0.0057	48.43%, 0.0038	48.73%, 0.0039	<b>48.80%</b> , 0.0038	
300x300_15_30(18)	51.04%, 0.0060	51.95%, 0.0098	52.67%, 0.0085	53.05%, 0.0087	53.29%, 0.0083	
300x300_15_30(24)	55.94%, 0.0157	57.58%, 0.0162	58.94%, 0.0099	59.26%, 0.0100	59.56%, 0.0088	
NSGA-II ( $\overline{Hyp}$ %, <i>std.dev</i> )						
Test case	Evaluations (Stop condition)					
Instance (routers)	50 000	100 000	200 000	300 000	400 000	
100x100_15_30(2)	42.40%, 0.0029	42.45%, 0.0030	42.67%, 0.0002	42.69%, 0.0002	42.69%, 0.0002	
100x100_15_30(3)	55.02%, 0.0044	55.37%, 0.0023	55.53%, 0.0027	55.55%, 0.0024	55.66%, 0.0004	
100x100_15_60(2)	31.60%, 0.0019	31.80%, 0.0012	31.82%, 0.0014	31.86%, 0.0010	31.94%, 0.0000	
100x100_15_60(3)	58.97%, 0.0035	59.25%, 0.0035	59.69%, 0.0019	59.73%, 0.0022	59.91%, 0.0016	
200x200_15_30(2)	34.39%, 0.0193	35.39%, 0.0175	36.33%, 0.0152	37.06%, 0.0103	37.42%, 0.0050	
200x200_15_30(4)	43.41%, 0.0435	44.46%, 0.0454	46.45%, 0.0509	47.22%, 0.0519	47.48%, 0.0537	
200x200_15_30(6)	53.70%, 0.0792	59.27%, 0.0643	64.61%, 0.0087	65.01%, 0.0094	65.26%, 0.0110	
200x200_15_30(9)	73.67%, 0.0173	76.16%, 0.0114	77.56%, 0.0117	78.14%, 0.0117	78.47%, 0.0109	
200x200_15_60(2)	23.40%, 0.0048	23.92%, 0.0046	23.93%, 0.0037	23.94%, 0.0028	24.00%, 0.0032	
200x200_15_60(4)	58.16%, 0.0086	58.72%, 0.0070	59.94%, 0.0056	60.08%, 0.0046	60.15%, 0.0045	
200x200_15_60(6)	71.79%, 0.0098	74.11%, 0.0076	75.42%, 0.0093	75.93%, 0.0103	76.37%, 0.0096	
200x200_15_60(9)	85.98%, 0.0105	88.38%, 0.0086	90.05%, 0.0074	90.51%, 0.0079	90.94%, 0.0079	
300x300_15_30(6)	38.22%, 0.0114	39.41%, 0.0086	40.31%, 0.0094	40.77%, 0.0097	41.11%, 0.0102	
300x300_15_30(12)	44.50%, 0.0148	46.39%, 0.0116	47.87%, 0.0105	48.11%, 0.0123	48.71%, 0.0102	
300x300_15_30(18)	47.01%, 0.0242	50.02%, 0.0209	52.53%, 0.0184	53.39%, 0.0167	<b>54.10%</b> , 0.0201	
300x300_15_30(24)	48.00%, 0.0213	52.52%, 0.0187	56.67%, 0.0363	58.86%, 0.0401	59.99%, 0.0419	
SPEA2 ( $\overline{Hyp}$ %, <i>std.dev</i> )						
Test case	Evaluations (Stop condition)					
Instance (routers)	50 000	100 000	200 000	300 000	400 000	
100x100_15_30(2)	42.11%, 0.0036	42.45%, 0.0028	42.65%, 0.0004	42.66%, 0.0002	42.66%, 0.0001	
100x100_15_30(3)	53.71%, 0.0085	53.78%, 0.0080	53.94%, 0.0072	54.24%, 0.0067	53.87%, 0.0065	
100x100_15_60(2)	31.35%, 0.0027	31.57%, 0.0018	31.78%, 0.0014	31.82%, 0.0012	31.87%, 0.0008	
100x100_15_60(3)	57.96%, 0.0068	58.63%, 0.0056	58.97%, 0.0078	58.84%, 0.0072	59.36%, 0.0041	
200x200_15_30(2)	34.08%, 0.0139	34.35%, 0.0137	34.72%, 0.0140	34.85%, 0.0113	34.99%, 0.0109	
200x200_15_30(4)	44.14%, 0.0328	44.71%, 0.0383	45.31%, 0.0382	45.49%, 0.0385	45.63%, 0.0385	
200x200_15_30(6)	57.93%, 0.0139	61.62%, 0.0160	63.22%, 0.0188	63.80%, 0.0164	64.07%, 0.0165	
200x200_15_30(9)	70.79%, 0.0075	74.49%, 0.0101	75.63%, 0.0106	75.97%, 0.0118	76.35%, 0.0111	
200x200_15_60(2)	22.91%, 0.0036	23.29%, 0.0036	23.65%, 0.0030	23.81%, 0.0024	23.94%, 0.0017	
200x200_15_60(4)	57.43%, 0.0074	58.57%, 0.0061	59.30%, 0.0043	59.74%, 0.0051	59.98%, 0.0033	
200x200_15_60(6)	71.24%, 0.0067	72.69%, 0.0060	73.59%, 0.0076	74.04%, 0.0087	74.35%, 0.0112	
200x200_15_60(9)	84.22%, 0.0050	86.45%, 0.0037	88.19%, 0.0096	89.27%, 0.0102	89.71%, 0.0102	
300x300_15_30(6)	39.13%, 0.0109	40.15%, 0.0105	40.89%, 0.0114	41.12%, 0.0118	41.21%, 0.0117	
300x300_15_30(12)	45.56%, 0.0145	47.03%, 0.0120	47.99%, 0.0101	48.46%, 0.0070	48.63%, 0.0083	
300x300_15_30(18)	48.67%, 0.0135	50.89%, 0.0084	52.11%, 0.0067	52.96%, 0.0092	53.19%, 0.0106	
300x300_15_30(24)	52.39%, 0.0277	56.93%, 0.0335	59.44%, 0.0345	60.34%, 0.0340	<b>60.86%</b> , 0.0344	

**Table 4.** P-values of the Wilcoxon–Mann–Whitney’s test comparing among algorithms using hypervolume metric

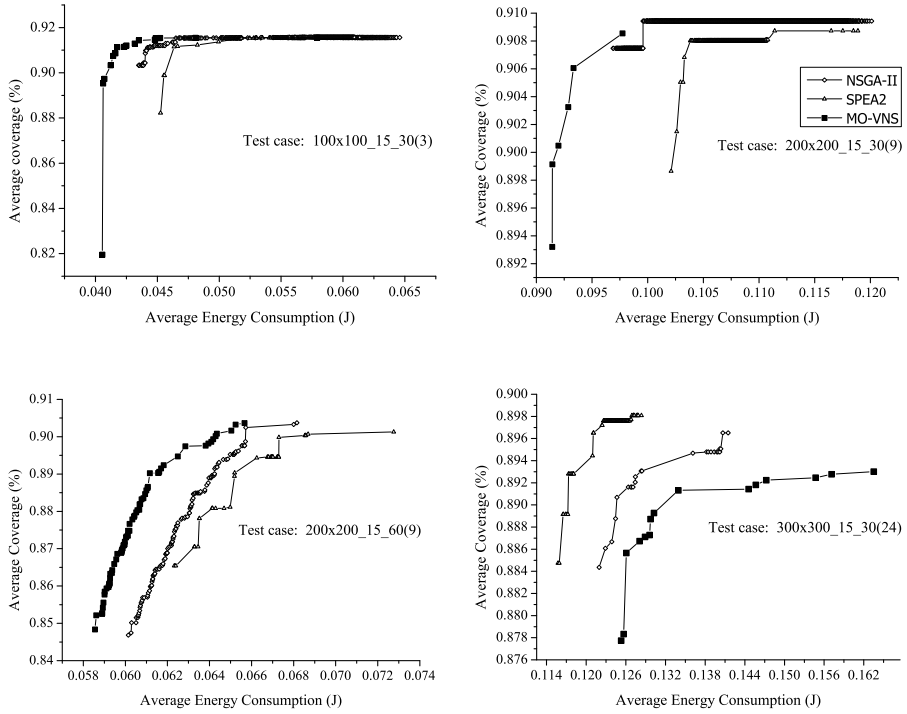
Instance (routers)	<i>MO-VNS vs NSGA-II</i>					<i>NSGA-II vs SPEA2</i>				
	50 000	100 000	200 000	300 000	400 000	50 000	100 000	200 000	300 000	400 000
100x100_15_30(2)	0.5791	0.2518	0.0000	0.0000	0.0000	0.0002	0.1520	0.0027	0.0000	0.0000
100x100_15_30(3)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
100x100_15_60(2)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0466	0.0163	0.0000
100x100_15_60(3)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
200x200_15_30(2)	0.0765	0.0023	0.0004	0.0003	0.0000	0.2014	0.0034	0.0000	0.0000	0.0000
200x200_15_30(4)	0.0000	0.0000	0.0000	0.0000	0.0000	0.9235	0.7157	0.2138	0.0388	0.0106
200x200_15_30(6)	0.0000	0.0000	0.0000	0.0000	0.0000	0.8080	0.2904	0.0008	0.0017	0.0018
200x200_15_30(9)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
200x200_15_60(2)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0075	0.0045	0.0075	0.0015	0.0009
200x200_15_60(4)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0075	0.0005	0.0001	0.0002	0.0001
200x200_15_60(6)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0075	0.0000	0.0000	0.0000	0.0000
200x200_15_60(9)	0.0000	0.0000	0.0000	0.0002	0.0133	0.0000	0.0000	0.0000	0.0000	0.0000
300x300_15_30(6)	0.0000	0.0000	0.0000	0.0000	0.0001	0.9978	0.9974	0.9669	0.8829	0.6402
300x300_15_30(12)	0.0000	0.0000	0.0114	0.0241	0.5310	0.9926	0.9873	0.7304	0.7715	0.3206
300x300_15_30(18)	0.0000	0.0000	0.4705	0.8303	0.9594	0.9975	0.9116	0.0916	0.1386	0.0215
300x300_15_30(24)	0.0000	0.0000	0.0077	0.2299	0.5913	1.0000	1.0000	0.9993	0.9235	0.7658

Instance (routers)	<i>MO-VNS vs SPEA2</i>					<i>SUMMARY</i>				
	50 000	100 000	200 000	300 000	400 000	50 000	100 000	200 000	300 000	400 000
100x100_15_30(2)	0.4151	0.2518	0.0000	0.0000	0.0000	—	—	MO-VNS	MO-VNS	MO-VNS
100x100_15_30(3)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
100x100_15_60(2)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
100x100_15_60(3)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_30(2)	0.0686	0.0008	0.0001	0.0001	0.0000	—	—	MO-VNS	MO-VNS	MO-VNS
200x200_15_30(4)	0.0002	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_30(6)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_30(9)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_60(2)	0.0052	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_60(4)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_60(6)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
200x200_15_60(9)	0.0000	0.0000	0.0000	0.0000	0.0000	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
300x300_15_30(6)	0.0000	0.0001	0.0003	0.0003	0.0002	MO-VNS	MO-VNS	MO-VNS	MO-VNS	MO-VNS
300x300_15_30(12)	0.0000	0.0004	0.1003	0.0761	0.2946	MO-VNS	MO-VNS	MO-VNS	MO-VNS	—
300x300_15_30(18)	0.0000	0.0000	0.0079	0.4967	0.4429	MO-VNS	MO-VNS	—	—	NSGA-II
300x300_15_30(24)	0.0000	0.3811	0.8258	0.9235	0.9596	MO-VNS	MO-VNS	MO-VNS	—	—

**Table 5.** Set coverage among algorithms  $C(A,B)$  for 400 000 evaluations

Instance (routers)	A		NSGA-II		SPEA2	
	MO-VNS	NSGA-II	MO-VNS	SPEA2	MO-VNS	NSGA-II
100x100_15_30(2)	100.00%	100.00%	0.00%	96.49%	0.00%	72.97%
100x100_15_30(3)	15.77%	0.63%	5.88%	100.00%	5.88%	50.50%
100x100_15_60(2)	94.80%	99.88%	0.00%	91.87%	0.00%	80.80%
100x100_15_60(3)	100.00%	100.00%	0.00%	74.60%	0.00%	39.80%
200x200_15_30(2)	33.76%	92.60%	0.00%	100.00%	0.00%	20.81%
200x200_15_30(4)	0.00%	82.79%	33.33%	100.00%	0.00%	0.00%
200x200_15_30(6)	100.00%	100.00%	0.00%	100.00%	0.00%	0.00%
200x200_15_30(9)	3.41%	99.83%	0.00%	100.00%	0.00%	0.00%
200x200_15_60(2)	100.00%	90.91%	0.00%	54.55%	0.00%	67.39%
200x200_15_60(4)	100.00%	91.67%	0.00%	0.00%	0.00%	47.06%
200x200_15_60(6)	80.37%	57.14%	0.00%	28.57%	0.00%	41.12%
200x200_15_60(9)	99.21%	100.00%	0.00%	100.00%	0.00%	0.00%
300x300_15_30(6)	100.00%	100.00%	0.00%	0.15%	0.00%	87.05%
300x300_15_30(12)	53.42%	9.51%	50.00%	4.94%	40.00%	92.82%
300x300_15_30(18)	0.00%	11.04%	75.00%	100.00%	62.50%	0.00%
300x300_15_30(24)	0.00%	0.00%	100.00%	0.00%	100.00%	100.00%
<b>Partial average</b>	<b>61.30%</b>	<b>71.00%</b>	<b>16.51%</b>	<b>65.70%</b>	<b>13.02%</b>	<b>43.77%</b>
<b>Average</b>	<b>66.15%</b>	<b>41.11%</b>	<b>28.40%</b>			



**Fig. 2.** Comparison between the median Pareto fronts obtained from NSGA-II, SPEA-2 and MO-VNS for 400 000 evaluations

value for the hypervolume. Next, we study whether differences among the average hypervolumes shown in Table 3 are significant. With this purpose, *Wilcoxon-Mann-Whitney's* test [11] (samples do not follow a normal distribution and are independent) was used with the following hypothesis: ( $H_0$ )  $Me_i$  is worse than  $Me_j$ , and ( $H_1$ )  $Me_i$  is better than  $Me_j$ , with  $i = 1, 2, 3$ ,  $j = 2, 3$ ,  $i < j$ , 1=MO-VNS, 2=NSGA-II and 3=SPEA2. The P-values obtained are shown in Table 4, where values greater than 0.05 (differences are not significant) are shaded. Thus, we can observe that MO-VNS provides the best behavior for experimental instances  $100 \times 100$  and  $200 \times 200$ , but in the most complex scenario the behavior is not so clear: MO-VNS provides best solutions with 6 routers and NSGA-II with 18, and in the other cases, differences are not significant.

On the other hand, the set coverage measure was used. We compare the coverage relation among median fronts from the 31 independent runs previously carried out for 400 000 evaluations. Observing Table 5, we reach the same conclusions as for hypervolume: MO-VNS provides the best results on average, but in the most complex instances the behavior is different: as the number of routers increases, quality results seem to get worse. Following this line, Fig. 2 graphically compares among median Pareto fronts obtained in some representatives test cases. In this figure, we can see the number of solutions obtained and its

distribution in the objective space. If we compare these solutions with the values of HO-AEC and HO-AC shown in Table 1 (AEC and AC values of each instance without using relay nodes), we check as adding routers in traditional WSNs is a good way to optimize them; for example, in  $200 \times 200_{15\_60(9)}$  a extreme solution of the MO-VNS front has a AEC of 0.0585, being its HO-AEC value 0.3871, i.e the value decreased by up 6.6 times with 9 routers.

## 5 Conclusions and Future Work

In this paper, we study the efficient network deployment by means of the inclusion of relay nodes to previously-established static traditional WSNs, that is the so-called Relay Node Placement Problem. With the aim of solving this problem, three different metaheuristics were used, two of them are genetic algorithms and the third is a multiobjective version of the trajectory algorithm VNS. We defined 16 tests cases to study the behavior of the algorithms, which we provide in [8]. As this is a multiobjective problem, two well-known multiobjective metrics were used (set coverage and hypervolume), which allow other authors to compare their approaches with ours. After analyzing the results by means of a widely used statistical methodology, we conclude that MO-VNS provides better results on average than both NSGA-II and SPEA2, in medium and small instances, but in more complex instances the behavior is different: if the number of routers used increases, quality results seem to get worse. In addition, we checked as the inclusion of relay nodes is a good way to optimize these networks.

As future line of research, it would be very interesting to use other metaheuristics with the aim of getting a framework that provides the best results in a wide range of situations. In addition, it would be useful to increase the number of test cases as well as the use of real scenarios to validate our hypothesis.

**Acknowledgments.** This work was partially funded by the Spanish Ministry of Economy and Competitiveness and the ERDF (European Regional Development Fund), under the contract TIN2012-30685 (BIO project), and by the Government of Extremadura, with the aid GR10025 to the group TIC015.

## References

1. Cardei, M., Du, D.Z.: Improving wireless sensor network lifetime through power aware organization. *Wireless Networks* 11, 333–340 (2005)
2. Cheng, X., Narahari, B., Simha, R., Cheng, M., Liu, D.: Strong minimum energy topology in wireless sensor networks: Np-completeness and heuristics. *IEEE Transactions on Mobile Computing* 2, 248–256 (2003)
3. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast elitist multi-objective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation* 6, 182–197 (2000)
4. Geiger, M.J.: Randomised variable neighbourhood search for multi objective optimisation. *Proceedings of the 4th EU/ME Workshop 0809.0271*, 34–42 (2008)

5. Han, X., Cao, X., Lloyd, E.L., Shen, C.C.: Fault-tolerant relay node placement in heterogeneous wireless sensor networks. *IEEE Transactions on Mobile Computing* 9, 643–656 (2010)
6. Konstantinidis, A., Yang, K., Zhang, Q.: An evolutionary algorithm to a multi-objective deployment and power assignment problem in wireless sensor networks. In: *Proceedings of IEEE GLOBECOM*, pp. 1–6 (2008)
7. Konstantinidis, A., Yang, K.: Multi-objective k-connected deployment and power assignment in wsns using a problem-specific constrained evolutionary algorithm based on decomposition. *Computer Communications* 34, 83–98 (2011)
8. Lanza-Gutiérrez, J.M., Gomez-Pulido, J.A., Vega-Rodríguez, M.A.: Instance sets for optimization in wireless sensor networks (2011), <http://arco.unex.es/wsnopt>
9. Lanza-Gutiérrez, J.M., Gómez-Pulido, J.A., Vega-Rodríguez, M.A., Sánchez-Pérez, J.M.: Relay node positioning in wireless sensor networks by means of evolutionary techniques. In: Kamel, M., Karray, F., Hagrass, H. (eds.) *AIS 2012. LNCS*, vol. 7326, pp. 18–25. Springer, Heidelberg (2012)
10. Lilliefors, H.W.: On the kolmogorov-smirnov test for normality with mean and variance unknown. *Journal of the American Statistical Association* 62, 399–402 (1967)
11. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics* 1, 50–60 (1947)
12. Martins, F., Carrano, E., Wanner, E., Takahashi, R., Mateus, G.: A hybrid multi-objective evolutionary approach for improving the performance of wireless sensor networks. *IEEE Sensors Journal* 11, 545–554 (2011)
13. Mukherjee, J.Y.B., Ghosal, D.: Wireless sensor network survey. *Computer Networks* 52, 2292–2330 (2008)
14. Perez, A., Labrador, M., Wightman, P.: A multiobjective approach to the relay placement problem in wsns. In: *Proceedings of IEEE WCNC*, vol. 1, pp. 475–480 (2011)
15. Shapiro, S.S., Wilk, M.B.: An analysis of variance test for normality (complete samples). *Biometrika* 52, 591–611 (1965)
16. Wang, Q., Xu, K., Takahara, G., Hassanein, H.: Device placement for heterogeneous wireless sensor networks: Minimum cost with lifetime constraints. *IEEE Transactions on Wireless Communications* 6, 2444–2453 (2007)
17. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation* 3, 257–271 (1999)
18. Zitzler, E.: *Evolutionary algorithms for multiobjective optimization: Methods and applications* (1999)
19. Zitzler, E., Laumanns, M., Thiele, L.: *Spea2: Improving the strength pareto evolutionary algorithm*. Tech. rep., Computer Engineering and Networks Laboratory (TIK), ETH Zurich (2001)

# Using Dynamic, Full Cache Locking and Genetic Algorithms for Cache Size Minimization in Multitasking, Preemptive, Real-Time Systems

Antonio Martí Campoy, Francisco Rodríguez-Ballester, and Rafael Ors Carot

Universitat Politècnica de València,  
Departamento de Informática de Sistemas y Computadores, 46022 València, Spain

**Abstract.** Cache locking have shown during the last years their usefulness easing the schedulability analysis of multitasking, preemptive, real-time systems. Cache locking provides a high degree of predictability while system performance is maintained at a similar level to that provided by regular, highly unpredictable, non-locked cache. Cache locking may also be useful to reduce hardware costs by means of reducing the size of the cache memory needed to make a real-time system schedulable. This work shows how full, dynamic cache locking may help to reduce the size of the cache memory versus a regular cache. This reduction is possible thanks to a genetic algorithm that selects the set of instructions that have to be locked in cache to provide the maximum cache size minimization while keeping the system schedulable.

**Keywords:** Genetic algorithm, Real-Time Systems, Cache Locking, schedulability analysis, cost-saving.

## 1 Introduction

Cache memories are an important advance in computer architecture, providing significant performance improvements. However, in the area of real-time systems, the use of cache memories introduces serious problems regarding predictability. The dynamic and adaptive behaviour of a cache memory reduces the average access time to main memory, but presents a non-deterministic fetching time [7]. In multitasking, preemptive, real-time systems, estimating the Worst Case Response Time (WCRT) of every task in a system populated with a regular cache becomes a problem with a solution hard to find due to the interference on the cache contents produced among the tasks.

In recent years, the use of locked caches have appeared as a solution to ease the schedulability analysis of multitasking, preemptive, real-time systems maintaining, at the same time, similar performance improvements than systems populated with regular cache memories. Several works have been presented to apply cache locking in real-time systems, both for instructions [3][15][14][1] and data [16]. In this work, we focus on instruction caches only, because 75% of accesses to main memory are to fetch instructions [7].

The paper is organised as follows: section two describes the objective and the goal of this work. Section three brief introduces the use of cache locking in multitasking, real-time systems. Section four describes with detail the Genetic Algorithm that selects the contents to load and lock in cache. Sections five, six and seven show the setup, the procedures and the results of experiments carried out. Finally, conclusions are presented in section eight.

## 2 Rationale

The ability to implement custom processors in FPGAs or ASICs has partly simplified the choice of the processor for a real-time system [11]. With these it is possible to build a system with its performance tailored to the actual requirements, decreasing the cost of the resulting system. What's more important, the designer can incorporate those architectural improvements that interest he most.

Although achieving predictability and easing the schedulability analysis is the main goal and advantage of cache locking, it may help to minimize the cache size thus saving costs and reducing the power consumption of the system.

The goal of this work is to show that LSM-dynamic use of cache locking may reduce the cache size needed to make a real-time system schedulable in front of a regular cache. And this advantage is added to the predictability and easiness of analysis provided by cache locking.

## 3 Use of Cache Locking

Cache locking means to keep all or part of cache contents away from replacement policy, or to allow replacement only in particular scenarios. In brief, the adaptive behaviour of regular caches is reduced or completely eliminated.

When the entire cache is locked, it is called full locking. If only some parts of the cache contents are locked, no matter if these parts are cache ways or cache lines, it is called partial locking. [6]

In multitasking systems, when locked contents remain unchanged for the whole life of the system, it is called static use or static cache locking.

If locked contents may change in a controlled way, for example in task switch, it is called dynamic use of cache locking. Some authors prefer to call this multiplexed or time-shared cache.

Several previous works in the literature apply cache locking over individual, isolated tasks in order to improve or accurately estimate the Worst-Case Execution Time (WCET) of a task. The work developed in this paper deals with multitasking, preemptive, real-time systems that presents a more complex problem, because includes the WCET estimation together with the WCRT estimation, dealing with intra-task and inter-task interference related to cache [2][8].

### 3.1 Full, Dynamic Use of Cache Locking

In here called dynamic use of cache locking every task may use and lock the cache in full for its own instructions. Replacement of cache contents is allowed,



but at some moments only. To our best knowledge, the first proposal of dynamic use of cache locking appears in [3] where cache contents may change only when a task starts its execution or resumes after a preemption, flushing the cache contents and reloading it with its own set of selected instructions. This is the behaviour considered in this work.

WCET estimation for each task in the system is easy, because during the execution of the task the cache contents is always the same, even after a preemption, because the first action when a task resumes its execution is to reload the cache with its own instructions.

Cache Response Time Analysis (CRTA) [2] is here used to estimate the response time of tasks, taking into account the time needed to reload the cache before a task runs or resumes its execution. This time, called cache refill penalty or cache-related preemption delay is added to the execution time of each task for each run and for each preemption, giving a safe upper bound of its WCRT.

Reloading cache contents when a task starts or resumes execution may be accomplished by software or hardware means. By software, the scheduler is overloaded with a routine that read from main memory the list of main memory blocks (instructions) that were off-line selected to load and lock in cache. For each main memory block, the routine accesses main memory two times: first, to fetch the address of the block to be loaded, and second, to effectively read this block and transfer it to the cache memory by means of specific instructions. Some other constant overload applies, and the total temporal cost to run this routine is added to CRTA as the cache refill penalty.

Loading instructions by means of software adds an important overhead, so in [15] a hardware solution is proposed. An extra, dedicated memory is added, called the Locking State Memory (LSM). Its role is to store the status of every main memory block, that is, whether it is selected or not to be locked in cache. This way the LSM provides a mechanism to discriminate which blocks must be loaded into the cache and hence a way to allow for automatic, on-demand loading of the selected main memory blocks. In other words, instead of locking the selected blocks into a locked cache, the same effect can be attained by avoiding loading into the cache the unselected blocks.

## 4 Using a Genetic Algorithm to Select Cache Contents

Cache locking intrinsically provides predictability. But other aspects, like performance, depend on the set of instructions selected to be locked in cache. Random selection of these instructions will give predictability and easiness of analysis, but system performance may be seriously degraded.

Several algorithms [5] have been developed in order to select instructions to load and lock in cache that improves system performance, or at least, maintains the system performance close to a system populated with a regular, non-predictable cache memory.

In [9] a new version of a genetic algorithm is proposed. The target of this algorithm is not to improve system performance but to reduce the size of the

cache memory needed to get an schedulable real-time system. This algorithm is used on static locking and outperforms previous results [4] where a performance-targeted algorithm was used.

In this paper the algorithm is adapted to be used on dynamic locking. This adaptation is done primarily in two points: in the representation of the problem in order to allow each task to fully utilize the cache, and in the evaluation of individuals where the CRTA equation corresponding to the dynamic cache locking using an LSM is used. The GA is executed in the design phase of the real-time system, so its execution time does not affect nor results in overhead for the real-time system. Below the main operators of the GA are described.

**Problem Representation.** A tri-dimensional matrix stores the status of main memory blocks in order to determine whether they are selected or not to be loaded and locked into cache. The first index used to access this matrix identifies to which task the memory block belongs, the second index identifies the set or cache memory line in which the block is mapped, and the third dimension is used to store the list of blocks that are mapped to that cache set/line together with their corresponding status. This rather complex structure allows crossing two individuals and mutating the resulting individuals guaranteeing that the cache mapping function will not be violated irrespective of the cache associativity degree. In fact, the representation is a list (of tasks) of lists (of cache sets/lines) of lists (of selected blocks for each cache set/line). The last list (selected blocks) is the responsible of fitting the mapping function. When individuals are created, the number of selected blocks for each set/line is at most the cache associativity degree. In crossover, the splitting point is restricted to the second dimension, avoiding the division of the third dimension of an individual, so the number of selected blocks remain unchanged and in a valid range. Regarding mutation, it only unlocks blocks, so it is not possible that the resulting number of selected blocks will be over the associativity degree.

**Initial Population.** The initial individuals are created by selecting a pseudo-random number of blocks to load and lock in cache. These blocks may belong to any task. Also, initialization creates a single individual with all of the blocks of every task marked as selected for being loaded and locked in cache memory. The purpose of including the special individual is to broaden the initial search space and guarantee that the algorithm starts with at least one schedulable individual.

The fitness function results from the combination of the result of the schedulability test, the number of used cache lines, and the system global utilisation. This way, the fitness value for an individual is a 3-tuple  $(S, L, U)$ , where  $S$  is the boolean result from the schedulability test,  $L$  is the cache size, measured in cache lines, the individual needs, and  $U$  is the global utilisation of the system. An individual  $i$  with fitness  $(S_i, L_i, U_i)$  is better than individual  $j$  with fitness  $(S_j, L_j, U_j)$  if:

- i)  $S_i$  and  $\text{not}(S_j)$  or*
- ii)  $S_i$  and  $S_j$  and  $(L_i < L_j)$  or*
- iii)  $S_i$  and  $S_j$  and  $(L_i = L_j)$  and  $(U_i < U_j)$*

That is, and individual is better than other if it is schedulable and the other one is not, or if it uses a smaller cache (in case both are achedulable), or, in case of a tie, it presents a lower system utilisation.

As selection policy binary tournament has been chosen. To select a parent two individuals are pseudo-randomly chosen, their fitness functions compared, and the better individual is chosen. The same procedure is used to choose the other parent. Besides this selection, an elitist selection is introduced. Two copies of the best individual are made, one of the copies is exposed to mutation while the other one suffers no mutation and is effectively incorporated into the next generation without any modification.

Single point crossover is used in this algorithm. The single point crossover is obtained by splitting every parent at a locus given by a pair (*task\_number*, *set\_number*) and creating two new individuals by exchanging the parts of the two parents.

In the mutation process the algorithm pseudo-randomly selects a set of locked blocks and marks them as unlocked, thus decreasing the number of locked blocks and therefore, the cache size.

The GA execution finishes when a predetermined number of generations has been reached. This value and the remaining input parameters are detailed below: Population size: 200 individuals; Termination condition: 5000 generations; Crossover probability: 100%; Mutation probability: 8%; Runs for each GA experiment to avoid seed effects: 25. The average cache size of these runs is presented in this work.

The input parameters of the genetic algorithm are: Algebraic expressions for tasks' WCET estimation [3]; List of main memory blocks used by each task; Periods of tasks; Hit and miss times; Cache line / main memory block size; Cache mapping function.

The output of the GA includes: worst-case execution time and worst-case response time of each task, the minimum cache size that makes the system schedulable, and the list of main memory blocks selected to be loaded and locked in the cache.

It is important to recall that execution time and response time of tasks provided by the genetic algorithm are not simulated, but estimated by means of cache-aware analysis methods (CRTA).

## 5 Experiment Setup

In order to study the ability to reduce the cache size the real-time systems presented in [9] were used. In this set of systems there are 14 different task sets. Tasks are artificially created to stress the locking/regular cache scheme. Main parameters of each task are defined, like its size, the number and size of loops and their nesting level, number of if-then-else structures and their respective sizes. These parameters are fixed or randomly selected. Then, a simple software tool creates the assembly code. This code is MIPSr2000 compatible.

Table 1 shows the different sets of tasks, the number of tasks per set, the sum of task sizes, their average size, and the system utilisation when the system runs on a 4096 lines-size regular cache.

Task periods are manually adjusted to force different number of preemptions among the tasks and therefore setting the system utilisation in different values, creating four scenarios. In all four scenarios and for all task sets, task deadlines are equal to task's periods and priority is assigned by the Rate Monotonic policy (the shorter the period the higher the priority).

The tasks in the first scenario have large periods, much longer than their respective response times, so there are no preemptions among tasks and the resulting system utilisation is low: around 30% when the system runs on a regular cache larger than the sum of all tasks sizes, that is, when only compulsory misses may occur. This scenario is called Low Interference (LI).

The tasks in the second scenario have also large periods, but not very far from their response times, so some preemptions may occur. System utilisation in this case is around 60% when system runs on a regular cache larger than the sum of all tasks sizes, that is, when only compulsory misses may occur. This scenario is called Low Medium Interference (LMI).

In the third scenario the tasks have periods close to their respective response times so many preemptions occur, increasing response times and therefore, system utilisation. In this case the system utilisation is around 80% when system runs on a regular cache larger than the sum of all tasks sizes, that is, with compulsory misses only. This scenario is called Medium High Interference (MHI).

Task periods in the fourth scenario are very close to their response times, so the number of preemptions is large and the system utilisation grows up to 95% when system runs on a regular cache larger than the sum of all tasks sizes. This scenario is called High Interference (HI).

Table 1 shows the system utilisation for each set of tasks, and for each configuration of deadlines when the system runs on a regular cache larger than the system size.

A total of 56 systems = 14 (task sets) x 4 (periods sets) are evaluated under two architectures: regular caches, and full, dynamic cache locking by means of hardware LSM.

Using the same task set but different task periods allows assessing the behaviour of the architectures and algorithms in front of a wider range of cache size requirements: as the task interference increases a larger cache is necessary to keep the system schedulable.

Results for regular caches come from simulation, while results for LSM cache locking come from estimation of WCET and WCRT by means of CRTA. This way, actual response time of tasks when using regular caches may be longer, but never shorter. In the other hand, actual response time of tasks when using cache locking may be shorter, but never longer, because they are estimated using conservative approaches. Conventional cache benefit from this situation because optimistic values are used for regular cache in front of conservative values for cache locking.

**Table 1.** Main parameters of the set of tasks used in experiments, and system utilisation for each scenario

Task set	# of tasks	Total size (instructions)	Average size (instructions)	LI (%)	LMI (%)	MHI (%)	HI (%)
1	3	2565	855.0	34.8	62.2	78.7	96.1
2	4	3397	849.3	35.0	60.6	78.1	98.4
3	8	1640	205.0	43.9	61.5	79.0	93.1
4	8	1640	205.0	36.3	59.5	82.3	95.7
5	5	2124	424.8	38.0	58.6	76.1	97.5
6	4	3252	813.0	32.0	62.4	76.7	93.9
7	3	3678	1226.0	40.9	57.1	80.5	97.8
8	3	1923	641.0	34.5	61.2	78.0	91.5
9	5	3086	617.2	38.7	58.8	78.9	96.5
10	3	3602	1200.7	35.0	62.0	80.2	93.2
11	4	1904	476.0	32.4	61.7	81.8	95.2
12	3	2378	792.7	36.5	68.7	83.3	96.8
13	5	790	158.0	29.4	59.3	75.9	98.0
14	5	2146	429.2	28.2	60.8	82.5	92.0

Regarding cache characteristics, fetching an instruction from cache takes 1 cycle while fetching an instruction from main memory takes 10 cycles. The only mapping function considered for cache locking is direct mapping, because this one is the most restrictive and provides the worst performance for cache locking [3][13]. The maximum size of the cache is 4096 lines, where each cache line contains four words, and each word is four bytes long. The instruction size is four bytes also. A main memory block presents the same size than a cache line.

The system includes a prefetch buffer with the size of one cache-line in order to get advantage of spatial locality when the processor fetches a main-memory block not locked in cache.

## 6 Experiment Procedures

The goal of the experiments carried out is to find the minimum cache size required to keep the system schedulable. This minimum size depends on the characteristics of the software system (number and size of tasks, periods, deadlines) and the underlying architecture the system runs on (regular cache or LSM-based dynamic cache locking).

For regular cache, in order to find the minimum cache size needed to make the system schedulable, several simulations of the system execution have been accomplished varying the cache size, starting in 1 line and increasing one by one, finishing when the cache size is one line larger than the total sum of the sizes of the system tasks. It would be possible to stop the experiments when system becomes schedulable but we want to study if there are discontinuities in the size of the cache that makes the system schedulable. No discontinuities were found.

Simulations were accomplished using a modified version of SPIM, a MIPS2000 simulator [12]. Input parameters for simulation are: Code of tasks in assembly; Periods of tasks; Cache size; Mapping policy (direct, two-way and full associative were tested, choosing the best result for each system, that is, the smaller cache size, usually direct mapping); Hit and miss times (1 and 10 cycles); Cache line / main memory block size (four four-byte-size instructions).

The output from simulation includes: worst execution time of tasks, worst response time of tasks, system utilisation, and schedulability test result (yes/no).

For cache locking the minimum cache size is found by a direct search using the genetic algorithm previously described. The genetic algorithm uses cache-aware analysis methods to estimate the response time of task and decide if the system is schedulable for a particular cache size.

The experiment setup and procedures give us a total of 112 experiments (56 systems multiplied by two cache architectures under study).

## 7 Experiment Results

The first outcome from experiment results is completely unexpected. For 14 experiments, the genetic algorithm considering the LSM architecture is unable to find a cache size to make the system schedulable. Even using a cache larger than the sum of the sizes of all the tasks that belong to the system, the genetic is unable to find a solution. The 14 experiments with no solution belong to the set of systems with the highest degree of task interference and very high system utilisation (scenario HI).

But the systems are in fact schedulable because when using a regular cache, there is a cache size for each system that makes it schedulable. So the problem is not in the real-time systems.

The reason the GA fails in finding a schedulable-making cache size for the LSM architecture is because of an excessively conservative approach when estimating the cache refill penalty.

The proposed CRTA analysis that is embedded in the GA considers that in every preemption the cache is completely flushed, and when a task resumes its execution, the full set of instructions selected to lock in cache is reloaded, without regard about how many of these instructions are useful, that is, will be executed before the next preemption.

Actually, both for regular and LSM cache, after a preemption, only instructions that are fetched by the processor between two preemptions may produce a miss, increasing the cache refill penalty. The instructions executed between two preemptions may be all of those selected to be locked in cache, or only a small subset, depending on the task structure and the time between preemptions.

With the conservative analysis the cache refill penalty is well-known, constant, and safe. But may be excessively larger. And the larger the number of preemptions, the worse the overestimation. This is the reason this problem arise for the systems with a high degree of interference; tasks suffer a lot of preemptions.

In following results, for the 14 systems where the GA is not able to find a cache size, we will use a cache with as many lines as the sum of the sizes of all tasks of

**Table 2.** Main statistics for LREG - LLSM

Statistic	LI	LMI	MHI	HI
Average	+123.35	-37.71	-146,5	-951.0
Min	-59	-414	-1092,0	-2074
Max	+327	+76	+189	-54
Cases<0	2	5	10	14
Cases>0	12	9	4	0
Avg. Reduction	47%	-5%	-14%	-39%
P-value t-test	0.0039	0.387	0,118	0.00059
T-test answer to null hypothesis	Reject	Do not reject	Do not reject	Reject
P-value Wilcoxon test	0.0057	0.66	0,09	0.00109
Wilcoxon test answer to null hypothesis	Reject	Do not reject	Do not reject	Reject

the system, and manually disables the locking mechanism. All instructions will be loaded in cache when they are fetched by the processor, and they will remain in cache because of the size of cache makes no conflicts.

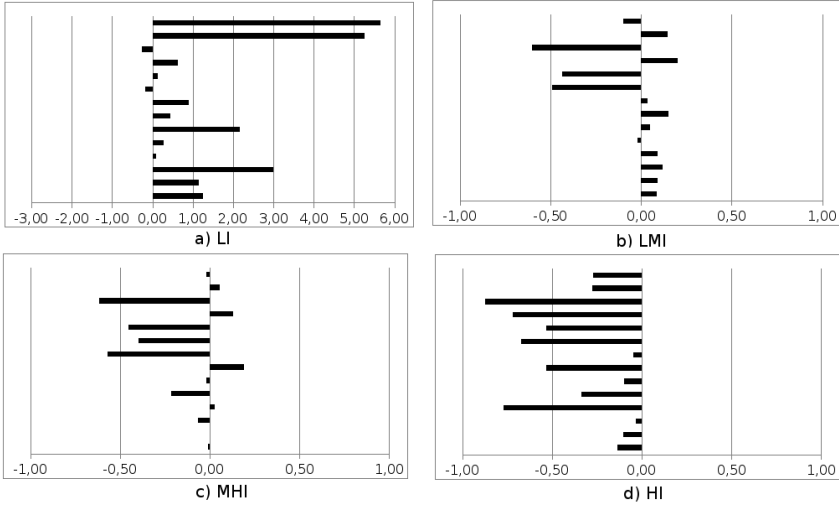
Analysis of result is accomplished grouping systems in the four previously described scenarios regarding its degree of task interference and value of utilisation (LI, LMI, MHI and HI). For each one of the four groups, a paired sample analysis is presented [10].

Table 2 shows the main statistics for LREG (number of cache lines used by the regular cache) minus LLSM (number of cache lines used by the LSM cache locking). for the four scenarios. The table also contains the p-value of a t-test and a Wilcoxon signed-rank test of null hypothesis and the answer to these tests, using an  $\alpha = 0.05$ .

Figure 1 shows the ratio LREG to LLSM for systems grouped in the four scenarios. The ratio is calculated as  $(LREG/LLSM) - 1$ . This figure shows if the LSM needs a smaller cache than the regular cache and the relative magnitude of cache saving. Values over 0 means the LSM needs less lines of cache than a regular cache. In the other hand, values below indicates that regular cache beats LSM, using a smaller cache to make schedulable a system.

In Figure 1-a) can be clearly observed that the LSM is able to make the systems schedulable using a smaller cache in most of the cases when the degree of interference between tasks is low (LI); this is confirmed by the answer of both tests which rejects the Null Hypothesis, meaning there is a statistically significant difference in the cache size needed by the LSM and the regular cache. The cache saving by means of using the LSM is in average of 45%, but in almost half of the cases the cache size reduction is over 100%.

In Figure 1-b), systems in scenario LMI, can be seen that there are more cases where the LSM needs a smaller cache, but in the cases where the LSM needs a bigger cache the cache size is much larger than the cache size needed by regular use. Attending the statistics in the third column (LMI) of table 2, the average



**Fig. 1.** Percentage of reduction of cache using LSM for experiments with periods a)LI, b)LMI, c)MHI and d)HI ( $LREG/LLSM) - 1$

cache size reduction is about -5%, that is, the LSM needs in average larger caches than the regular use of cache, but the answer of both tests is that the Null Hypothesis cannot be rejected; that is, there is no statistically significant differences between the cache sizes used by the two studied uses of cache.

In Figure 1-c), systems in scenario MHI, a slight bias to negative values can be appreciated; that is, there are more cases where the LSM needs a larger cache to make the system schedulable. Attending the statistics shown in the fourth column (MHI) of table 2, the LSM needs, in average, a 10% larger cache than the regular use of cache (reduction of -0.09%), but response of Null Hypothesis tests are do not reject; that is, there is no statistically significant difference between the cache size used by the two studied uses of cache.

Finally, the figure 1-d) shows the ratio  $LREG$  to  $LLSM$  for systems with High Interference, HI, calculated as  $(LREG/LLSM) - 1$ . The data in this figure is very clear: for all cases using a regular cache needs a smaller cache to make the system schedulable. And in front of LSM, the cache needed by the regular cache is about 30% smaller in average, and even more than 50% smaller in some cases. Statistics on the fifth column of table 2 confirms this conclusion, in particular the tests that reject the Null Hypothesis; that is, there are statistically significant differences between the regular use of cache and the LSM.

Results of the analysis of the four groups are coherent with the analysis of overestimation described in the beginning of this section. The larger the number of preemptions, the larger the overestimation. This way, the larger the interference between tasks, the smaller the cache saving by means of LSM dynamic use of cache locking. For one of the scenarios (LI) the LSM is able to significantly reduce the size of cache required to make the system schedulable. That is, using



cache locking by means of a LSM architecture helps the system designer to save costs and power. In the other extreme, another scenario (HI) shows that the LSM needs larger caches than the regular use, so its use is not recommended if the main concern is related to system costs or power consumption. For the two intermediate scenarios (LMI and MHI) there is a tie, that is, both the LSM and the regular cache will need the same size of cache to make the system schedulable. So, in general terms and talking about these intermediate scenarios, there is no significant difference in using a regular cache or a dynamically locked cache, concerning about system cost and size only.

But cache locking adds a precious value to the design process of a real-time system because they provide predictability and easiness of analysis.

## 8 Conclusions

This work presents a new application of dynamic cache locking, extending the research from a previous proposal where static cache locking was applied to reduce the instruction cache size in preemptive, multi-task, real-time systems. Results from experiments carried out show, first, that for some systems the dynamic use of cache locking is not suitable due to the overly conservative considerations that have to be used in the schedulability analysis, so for high interference, high utilisation systems, regular caches provide better results.

Second, there is a large number of cases where no statistically significant difference exists, in average, in the cache size needed to make a system schedulable when using a regular cache or a locked cache. In these cases, and for the same cache size, cache locking offers, in front of a regular cache, the benefit of high predictability and simplicity in the schedulability analysis.

Only when the system utilisation is low a LSM-based locked cache allows a notable significant cache size minimization maintaining the predictable behaviour and ease of analysis.

Experiment results also show that for some systems, and under some conditions, there may be important differences in the cache size needed to make the system schedulable. Future work will study the system characteristics and conditions that determine the architecture that gets a smaller cache, so recommendations to the system designer can be done.

**Acknowledgments.** This work is partially supported by PAID-06-11/2055 of Universitat Politècnica de València and TIN2011-28435-C03-01 of Ministerio de Ciencia e Innovación

## References

1. Aparicio, L.C., Segarra, J., Rodriguez, C., Vials, V.: Improving the wcet computation in the presence of a lockable instruction cache in multitasking real-time systems. *Journal of Systems Architecture* 57(7), 695–706 (2011), Special Issue on Worst-Case Execution-Time Analysis, <http://www.sciencedirect.com/science/article/pii/S1383762110001086>

2. Busquets-Mataix, J., Serrano, J., Ors, R., Gil, P., Wellings, A.: Adding instruction cache effect to schedulability analysis of preemptive real-time systems. In: Proceedings of the Real-Time Technology and Applications Symposium, pp. 204–212. IEEE (June 1996)
3. Campoy, A.M., Ivars, A.P., Mataix, J.V.B.: Dynamic use of locking caches in multitask, preemptive real-time systems. In: Proceedings of the 15th World Congress of the International Federation of Automatic Control (2002)
4. Campoy, A.M., Rodríguez-Ballester, F., Ors, R., Serrano, J.: Saving cache memory using a locking cache in real-time systems. In: Proceedings of the 2009 International Conference on Computer Design, pp. 184–189 (July 2009)
5. Campoy, A.M., Puaut, I., Ivars, A.P., Mataix, J.V.B.: Cache contents selection for statically-locked instruction caches: An algorithm comparison. In: Proceedings of the 17th Euromicro Conference on Real-Time Systems, pp. 49–56. IEEE Computer Society, Washington, DC (2005),  
<http://portal.acm.org/citation.cfm?id=1084012.1084148>
6. Ding, H., Liang, Y., Mitra, T.: Wcet-centric partial instruction cache locking. In: Proceedings of the 49th Annual Design Automation Conference, DAC 2012, pp. 412–420. ACM, New York (2012),  
<http://doi.acm.org/10.1145/2228360.2228434>
7. Hennessy, J.L., Patterson, D.A.: Computer Architecture: A Quantitative Approach, 4th edn. Morgan Kaufmann (2006)
8. Lee, C.G., Lee, K., Hahn, J., Seo, Y.M., Min, S.L., Ha, R., Hong, S., Park, C.Y., Lee, M., Kim, C.S.: Bounding cache-related preemption delay for real-time systems. *IEEE Transactions on Software Engineering* 27(9), 805–826 (2001)
9. Martí-Campoy, A., Rodríguez-Ballester, F., Tamura Morimitsu, E., Ors, R.: An algorithm for deciding minimal cache sizes in real-time systems. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO 2011, pp. 1163–1170. ACM, New York (2011),  
<http://doi.acm.org/10.1145/2001576.2001733>
10. McPherson, G.: Applying and Interpreting Statistics. A Comprehensive Guide, 2nd edn. Springer Texts in Statistics. Springer (2001)
11. Navabi, Z.: Embedded Core Design with FPGAs. McGraw-Hill Professional (2006)
12. Patterson, D., Hennessy, J.: Computer Organization and Design: The Hardware/software Interface. Morgan Kaufmann (1994)
13. Puaut, I., Pais, C.: Scratchpad memories vs locked caches in hard real-time systems: a quantitative comparison. In: Proceedings of the conference on Design, automation and test in Europe, DATE 2007, pp. 1484–1489. EDA Consortium, San Jose (2007),  
<http://dl.acm.org/citation.cfm?id=1266366.1266692>
14. Sascha Plazar, J.C.K., Marwedel, P.: Wcet-aware static locking of instruction caches. In: Proceedings of the 2012 International Symposium on Code Generation and Optimization, pp. 44–52 (2012)
15. Tamura, E., Busquets-Mataix, J., Campoy, A.M.: Towards predictable, high-performance memory hierarchies in fixed-priority preemptive multitasking real-time systems. In: Proceedings of the 15th International Conference on Real-Time and Network Systems (RTNS-2007), pp. 75–84 (2007)
16. Vera, X., Lisper, B., Xue, J.: Data cache locking for tight timing calculations. *ACM Trans. Embed. Comput. Syst.* 7(1), 4:1–4:38 (2007),  
<http://doi.acm.org/10.1145/1324969.1324973>

# General Quantum Encryption Scheme Based on Quantum Memory

Marius Nagy and Naya Nagy

College of Computer Engineering and Science  
Prince Mohammad Bin Fahd University  
Al Azeziya, Eastern Province  
Kingdom of Saudi Arabia  
{mnagy, nnagy}@pmu.edu.sa

**Abstract.** In cryptography, quantum information processing can be used to do much more than just key distribution. Simple quantum transformations augmented with the ability to store qubits in a quantum memory are the building blocks of a protocol allowing two parties to communicate secretly by encoding/decoding the exchanged message directly through quantum means, without the need to establish a secret encryption/decryption key first. Consequently, our quantum mechanical process of securely transmitting a message through a public channel is simpler, cleaner, faster and computationally more efficient than the two-step scenario with a quantum distributed classical key. The probability of catching a potential eavesdropper can be made arbitrarily large by increasing the length of the signature string attached to the message.

**Keywords:** quantum gates, quantum memory, measurement, cryptography, quantum protocol, security, eavesdropping, bit rank.

## 1 Introduction

Several techniques exist that exploit quantum effects for key distribution [1–4]. Regardless of whether they rely on quantum entanglement or not, all these quantum protocols are used with a single goal: to establish a classical secret key that is subsequently used to encrypt/decrypt a message using classical cryptographic algorithms. An important characteristic of the classical secret key is that it is *randomly* generated. No quantum key distribution scheme can be used to distribute a key that is known *a priori* to any of the communicating parties. This randomness comes from the implicit randomness associated with the measurement postulate in quantum mechanics.

In this paper, we develop a general quantum protocol for transmitting a classical message through a quantum channel without establishing a secret key first, but instead using simple unary quantum gates to directly encrypt/decrypt the message by quantum means. Certainly, at this point, the quantum computing technology is still in its infancy, but once it will mature, the computational power required by our protocol is quite limited: the ability to effect relatively

simple quantum transformations on single qubits. By comparison, the RSA algorithm [8] used to just distribute the secret key needed to encrypt/decrypt the actual message through classical means is much more computationally intensive. Adding the fact that establishing and distributing a cryptographic key is no longer needed in our method, the whole process of securely transmitting a message between two communicating parties is greatly simplified and streamlined.

Beside simple quantum gates, our scheme also relies on the use of a quantum memory capable of storing qubits (described by their quantum states) for a certain amount of time as detailed in the description of the protocol. Although building such a quantum memory in practice is a challenging endeavor, important steps in this direction have been recently reported [5, 10, 6, 9]. Any practical implementation of a quantum protocol aimed at securing communications has to find an appropriate physical embodiment for the qubits transmitted over the quantum channel. The best choice in this respect seems to be photons, whose polarizations can easily be manipulated and which are, by definition, very fast, traveling at the speed of light. On the other hand, photons are not well suited for storage, where solid-state approaches seem to be the most promising technology.

Now, two separate teams, one led by Wolfgang Tittel at the University of Calgary in Alberta, Canada [10] and another led by Nicolas Gisin at the University of Geneva in Switzerland [5] are reporting advances on the road to make the two technologies work together. Experimenting with different types of crystals, they managed to have the quantum state of a photon being captured in solid crystals through entanglement. Furthermore, scientists at Harvard University have developed a room-temperature quantum memory that can hold information on the order of seconds by using the spin on the nucleus of an atom inside a diamond to physically realize a qubit [6]. But the record on how long a superposition state can be maintained definitely belongs to a team led by Professor Mike Thewalt of Simon Fraser University, Canada [9]. Using the spins of atomic nuclei embedded in silicon, the research team were able to create a superposition state which lasted for 192 seconds (more than three minutes). These advances seem to hint to the possibility of practical realizations for protocols using quantum memories (like the one described in this paper) in the near future.

The remainder of the paper is organized as follows. Next section describes in detail each step of the quantum protocol that can be used to directly encrypt/decrypt a classical message. Sections 3 and 4 deal with the analysis of two common eavesdropping strategies: transparent eavesdropping and opaque eavesdropping. Possible variations of the protocol are discussed in Section 5. Finally, Section 6 presents some conclusions and summarizes the most important results of the paper.

## 2 Protocol Description

In order to communicate secretly, the two parties (commonly referred to as Alice and Bob) are assumed to have access to the following resources:

- a public quantum channel capable of delivering a block of qubits from Alice to Bob. This could be a fiber-optic cable or even air, depending on the

particular physical embodiment chosen for a qubit. No particular restrictions are imposed on the quantum channel. In particular, it is open to any form of eavesdropping.

- a public classical channel that allows Alice and Bob to communicate with each other, exchanging classical information. Although this channel is also public and open to eavesdropping, it is authenticated. This means that Alice has the certainty of speaking to (communicating with) Bob and Bob has the certainty of speaking to Alice.
- a quantum memory required by Bob to store the qubits sent by Alice until the signature is verified and they can be decrypted.
- the ability to perform quantum information processing, namely applying single-qubit gates like  $U_\theta$  and  $R_\varphi$  (see below) and measurements in the normal computational basis  $\{|0\rangle, |1\rangle\}$ . Note that this is not equivalent to the power of a general quantum computer, since two-qubit gates are required for universal quantum computation.

The information flow diagram describing the steps of the protocol is given in Fig. 1. The process starts with Alice choosing an encoding basis for the bitstring she wants to send to Bob. This bitstring is formed by the actual message to which a signature bitstring is appended. The signature will allow Alice and Bob to detect anyone trying to eavesdrop on the transmitted message. This means that every bit in the plain text block (message and signature) will be converted to one of the two basis vectors of the quantum basis chosen. An intuitive graphical representation of an encoding basis is a straight line going through the center of the Bloch sphere. Such a line is fully specified by two real-valued parameters: the angle  $\theta$  between the line and the  $z$  axis, and the angle  $\varphi$  between the projection of the line on the equatorial plane and the  $x$  axis (see Fig. 2). In some sense, this encoding method is somewhat similar to *anamorphosis*, since the actual message can only be "seen" when viewed from the proper angle. With the major difference that attempting to "read" the message from any angle except the correct one is equivalent to a quantum measurement that will necessarily alter the message.

Therefore, in order to choose an encoding basis, Alice needs to decide on a pair  $(\theta, \varphi)$ . For example, the pair  $(0, 0)$  specifies the computational basis, described by the two basis vectors  $|0\rangle$  and  $|1\rangle$  aligned along the  $z$  axis. Alternatively, pair  $(\pi/2, 0)$  specifies the Hadamard basis, described by basis vectors  $H|0\rangle = (1/\sqrt{2})|0\rangle + (1/\sqrt{2})|1\rangle$  and  $H|1\rangle = (1/\sqrt{2})|0\rangle - (1/\sqrt{2})|1\rangle$ , aligned along the  $x$  axis. As a last example, pair  $(\pi/2, \pi/2)$  specifies an encoding basis whose base vectors are  $(1/\sqrt{2})|0\rangle + (i/\sqrt{2})|1\rangle$  and  $(1/\sqrt{2})|0\rangle - (i/\sqrt{2})|1\rangle$ . Note that these two vectors are oriented in opposite directions along the  $y$  axis (see Fig. 3).

In general, for an arbitrarily chosen encoding basis  $(\theta, \varphi)$ , each 0 bit in the plaintext is embodied in a qubit with the quantum state

$$|\Psi_0\rangle = R_\varphi U_\theta |0\rangle = \cos \frac{\theta}{2} |0\rangle + e^{i\varphi} \sin \frac{\theta}{2} |1\rangle. \tag{1}$$

Similarly, a qubit in the quantum state

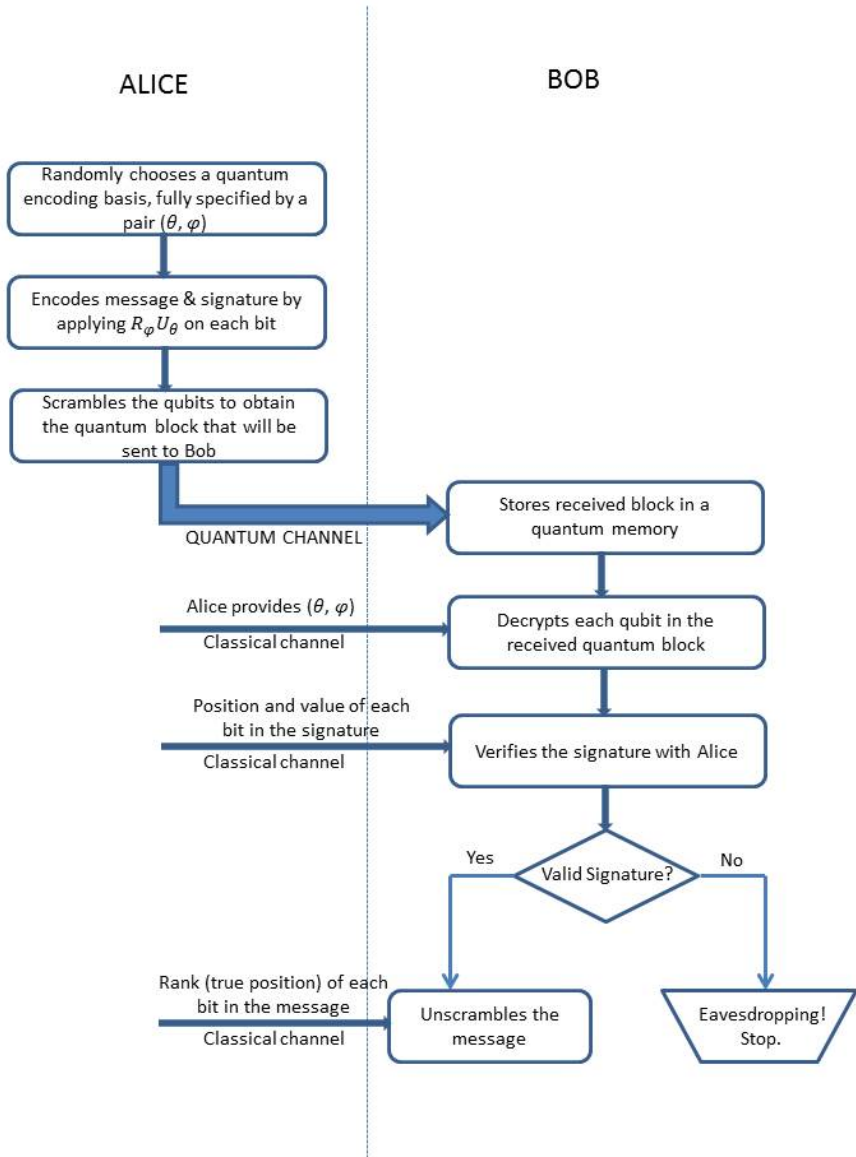
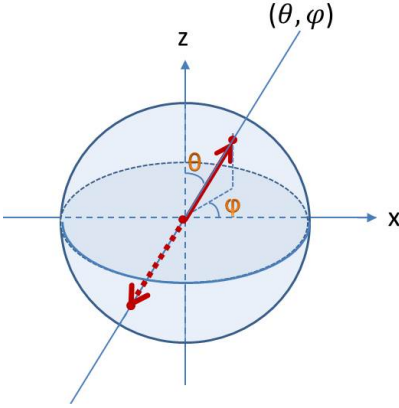
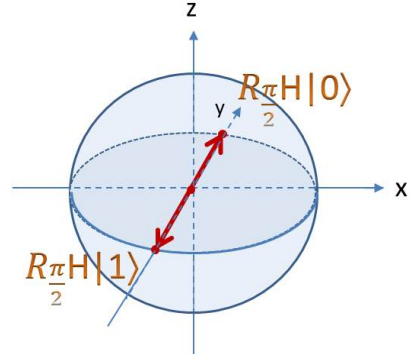


Fig. 1. Information flow diagram outlining the steps of the communication protocol



**Fig. 2.** Encoding basis is a straight line through the center of the Bloch sphere, specified by a pair  $(\theta, \varphi)$



**Fig. 3.** Pair  $(\frac{\pi}{2}, \frac{\pi}{2})$  specifies an encoding basis with base vectors  $\{\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle, \frac{1}{\sqrt{2}}|0\rangle - \frac{i}{\sqrt{2}}|1\rangle\}$

$$|\Psi_1\rangle = R_\varphi U_\theta |1\rangle = \sin \frac{\theta}{2} |0\rangle - e^{i\varphi} \cos \frac{\theta}{2} |1\rangle \tag{2}$$

will carry a value of 1. The quantum operations (gates)  $U_\theta$  and  $R_\varphi$  are described by the following matrices:

$$U_\theta = \begin{bmatrix} \cos \frac{\theta}{2} & \sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & -\cos \frac{\theta}{2} \end{bmatrix}; R_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}. \tag{3}$$

Once the plaintext is properly encoded, Alice will scramble the qubits so that signature qubits are scattered throughout and interspersed with message qubits. Consequently, the original ordering of bits in the message and in the signature (the rank of each bit) will no longer be preserved after the scrambling process. Whatever the scrambled quantum block is, Alice will send it through the quantum channel over to Bob, who will store each qubit received in a quantum memory, in the order they arrive. With this step, the quantum communication part of the protocol, that is, communication through the quantum channel is over. In the second phase, all communication between Alice and Bob is carried out through the public authenticated classical channel.

This second phase starts with Alice disclosing to Bob the exact encoding basis that she used in order to encrypt the plaintext. With this information Bob is able to decrypt each qubit (whether belonging to the message or to the signature) stored in his quantum memory. Next, Alice shares with Bob the position and value of each bit in the signature string. In this way, they can verify that the signature string received by Bob matches exactly the one sent by Alice. A perfect match will be taken as proof that the encoded quantum block was not tampered with while in transit, or at least not to a significant level. Of course, a potential

eavesdropper may remain undetected if she is lucky enough not to disturb the quantum states of the qubits in the signature. But this probability can be made arbitrarily small by increasing the signature length.

Finally, if the verification step reveals no trace of an eavesdropper, Alice informs Bob about the rank (actual position) of each bit in the message string. This allows Bob to re-arrange the bits in the proper order and thus recover the original message.

### 3 Analysis

In this section we analyze the security of the protocol by investigating the effects of a possible eavesdropping act on the quantum and classical communication channels. Since the quantum channel is public, an eavesdropper (conventionally named Eve) may choose to act on any of the qubits passing by on their way from Alice to Bob. For concreteness, but without loss of generality, let us analyze the scenario in which Eve eavesdrops on a qubit encoding the value 0. The other case, in which a qubit encodes a 1 is perfectly similar.

The quantum state of a qubit embodying a 0 is  $|\Psi_0\rangle$  (see eq. 1). When Eve intercepts this qubit, she may try to measure it directly in the computational basis or try to guess what the encoding basis may have been, so that she can decrypt the qubit before measuring it. Let us denote the encoding basis (the one chosen by Alice) by  $(\theta_A, \varphi_A)$  and label the basis guessed by Eve  $(\theta_E, \varphi_E)$ . Trying to decrypt the qubit, Eve applies  $R_{-\varphi_E}$  followed by  $U_{\theta_E}$  to  $|\Psi_0\rangle$  altering the qubit state as follows:

$$\begin{aligned} U_{\theta_E} R_{-\varphi_E} |\Psi_0\rangle &= U_{\theta_E} \left( \cos \frac{\theta_A}{2} |0\rangle + e^{i(\varphi_A - \varphi_E)} \sin \frac{\theta_A}{2} |1\rangle \right) \\ &= \left( \cos \frac{\theta_E}{2} \cos \frac{\theta_A}{2} + e^{i(\varphi_A - \varphi_E)} \sin \frac{\theta_E}{2} \sin \frac{\theta_A}{2} \right) |0\rangle \\ &\quad + \left( \sin \frac{\theta_E}{2} \cos \frac{\theta_A}{2} - e^{i(\varphi_A - \varphi_E)} \cos \frac{\theta_E}{2} \sin \frac{\theta_A}{2} \right) |1\rangle \end{aligned} \quad (4)$$

Consequently, Eve will now measure a 0 with probability

$$\begin{aligned} p_{Eve}^0 &= \left| \cos \frac{\theta_E}{2} \cos \frac{\theta_A}{2} + e^{i(\varphi_A - \varphi_E)} \sin \frac{\theta_E}{2} \sin \frac{\theta_A}{2} \right|^2 \\ &= \cos^2 \frac{\theta_A - \theta_E}{2} - \frac{1}{2} \sin \theta_E \sin \theta_A (1 - \cos \Delta\varphi) \end{aligned} \quad (5)$$

and a 1 with probability

$$\begin{aligned} p_{Eve}^1 &= \left| \sin \frac{\theta_E}{2} \cos \frac{\theta_A}{2} - e^{i(\varphi_A - \varphi_E)} \cos \frac{\theta_E}{2} \sin \frac{\theta_A}{2} \right|^2 \\ &= \sin^2 \frac{\theta_A + \theta_E}{2} - \frac{1}{2} \sin \theta_E \sin \theta_A (1 + \cos \Delta\varphi) \end{aligned} \quad (6)$$

where  $\Delta\varphi = \varphi_A - \varphi_E$ .



Eve is aware that her actions may have modified the state of the qubit, so before sending it further on to Bob, she will try to undo the consequences of her eavesdropping by applying  $U_{\theta_E}$  followed by  $R_{\varphi_E}$ . Therefore, what Bob receives (from Eve) is a qubit in the state

$$|\Phi_0\rangle = \cos \frac{\theta_E}{2} |0\rangle + e^{i\varphi_E} \sin \frac{\theta_E}{2} |1\rangle \tag{7}$$

with probability  $p_{Eve}^0$ , or in the state

$$|\Phi_1\rangle = \sin \frac{\theta_E}{2} |0\rangle - e^{i\varphi_E} \cos \frac{\theta_E}{2} |1\rangle \tag{8}$$

with probability  $p_{Eve}^1$ .

Assuming that the qubit comes straight from Alice, Bob now applies  $R_{-\varphi_A}$  and  $U_{\theta_A}$  to decode it:

$$\begin{aligned} U_{\theta_A} R_{-\varphi_A} |\Phi_0\rangle &= \left( \cos \frac{\theta_A}{2} \cos \frac{\theta_E}{2} + e^{i(\varphi_E - \varphi_A)} \sin \frac{\theta_A}{2} \sin \frac{\theta_E}{2} \right) |0\rangle \\ &+ \left( \sin \frac{\theta_A}{2} \cos \frac{\theta_E}{2} - e^{i(\varphi_E - \varphi_A)} \cos \frac{\theta_A}{2} \sin \frac{\theta_E}{2} \right) |1\rangle \end{aligned} \tag{9}$$

$$\begin{aligned} U_{\theta_A} R_{-\varphi_A} |\Phi_1\rangle &= \left( \cos \frac{\theta_A}{2} \sin \frac{\theta_E}{2} - e^{i(\varphi_E - \varphi_A)} \sin \frac{\theta_A}{2} \cos \frac{\theta_E}{2} \right) |0\rangle \\ &+ \left( \sin \frac{\theta_A}{2} \sin \frac{\theta_E}{2} + e^{i(\varphi_E - \varphi_A)} \cos \frac{\theta_A}{2} \cos \frac{\theta_E}{2} \right) |1\rangle \end{aligned} \tag{10}$$

When measuring  $U_{\theta_A} R_{-\varphi_A} |\Phi_0\rangle$ , Bob will obtain 0 with probability  $p_{Eve}^0$ . Similarly, if the state of the qubit received by Bob is  $|\Phi_1\rangle$ , he will measure a 0 with probability  $p_{Eve}^1$ . Overall, the probability that Bob correctly decodes the qubit and obtains a 0 is  $(p_{Eve}^0)^2 + (p_{Eve}^1)^2$ . In this case, Eve’s eavesdropping activity remains undetected. The probability of detection on a single qubit is therefore:

$$p_{d,1} = 1 - ((p_{Eve}^0)^2 + (p_{Eve}^1)^2) = 2p_{Eve}^0 p_{Eve}^1 = 2p_{Eve}^0 (1 - p_{Eve}^0) \tag{11}$$

This probability achieves its maximum of 1/2 when  $p_{Eve}^0 = p_{Eve}^1 = 1/2$ . This happens when the basis guessed by Eve  $(\theta_E, \varphi_E)$  is "maximally non-orthogonal" to the basis  $(\theta_A, \varphi_A)$  chosen by Alice. In BB84 [2] for example, this maximum non-orthogonality is realized by choosing horizontal/vertical together with diagonal polarization. In terms of the Bloch sphere representation, two bases are "maximally non-orthogonal" if the two straight lines corresponding to the two bases are perpendicular to each other. On the other hand, the detection probability is 0, when Eve chooses the same basis as Alice ( $p_{Eve}^0 = 1$ ) or an orthogonal basis in which the roles of the two basis vectors are reversed ( $p_{Eve}^0 = 0$ ).

When Alice uses a signature bitstring of length  $n$ , the probability of detecting the disruptions caused by eavesdropping on the qubits encoding the signature grows to

$$p_{d,n} = 1 - ((p_{Eve}^0)^2 + (p_{Eve}^1)^2)^n \tag{12}$$

Since  $(p_{Eve}^0)^2 + (p_{Eve}^1)^2 = 2(p_{Eve}^0)^2 - 2p_{Eve}^0 + 1 \in (0, 1]$ , it follows that

$$\lim_{n \rightarrow \infty} p_{d,n} = 1, \quad (13)$$

except for the particular case when Eve correctly guesses the encoding basis ( $p_{Eve}^0 = 1$  or  $p_{Eve}^1 = 1$ ) and remains undetected ( $p_{d,1} = p_{d,n} = 0$ ). Consequently, the longer the signature string is, the larger the number of qubits that are tested for eavesdropping and the higher the probability to catch a potential eavesdropper.

## 4 Opaque Eavesdropping

In the previous section, we have analyzed a rather elaborate scheme for Eve to hide her presence and make her eavesdropping actions transparent to Alice and Bob. Even with all those precautions, we have seen that the detection rate can be pushed as high as desired by increasing the number of bits tested for eavesdropping in the signature string. In this section, we investigate a more direct, opaque eavesdropping strategy in which Eve directly measures some or all of the qubits traveling through the quantum channel from Alice to Bob.

Again, without loss of generality, let us assume Eve intercepts a qubit encoding a value of 0. Such a qubit is described by quantum state  $|\Psi_0\rangle$  (see eq. 1). Upon measuring this qubit in the normal computational basis, Eve will observe a 0 with probability  $p_{Eve}^0 = \cos^2(\theta/2)$  and a 1 with probability  $p_{Eve}^1 = \sin^2(\theta/2)$ , where  $\theta$  is one of the two parameters characterizing the encoding basis chosen by Alice. According to the measurement postulate of quantum mechanics, the post-measurement state of the qubit must be compatible with the measurement outcome, so Eve will pass on to Bob a qubit in state  $|0\rangle$  (with probability  $p_{Eve}^0$ ) or a qubit in state  $|1\rangle$  (with probability  $p_{Eve}^1$ ).

During the second phase of the protocol, after Alice has disclosed the encoding basis, Bob can proceed to decrypt the received qubits. Note that at this time, although Eve can also eavesdrop on the classical communication channel and thus gain knowledge of  $\theta$  and  $\varphi$ , the message qubits are no longer in her possession, so there is nothing else she can do to increase her knowledge about the transmitted message. By applying  $R_{-\varphi}$  and  $U_\theta$  to the received qubit, Bob will evolve its quantum state to

$$U_\theta R_{-\varphi}|0\rangle = \cos\frac{\theta}{2}|0\rangle + \sin\frac{\theta}{2}|1\rangle \quad (14)$$

with probability  $p_{Eve}^0$ , or to

$$U_\theta R_{-\varphi}|1\rangle = U_\theta(e^{-i\varphi}|1\rangle) = e^{-i\varphi}(\sin\frac{\theta}{2}|0\rangle - \cos\frac{\theta}{2}|1\rangle) \quad (15)$$

with probability  $p_{Eve}^1$ . A measurement on these quantum states will yield a 0 (correct decoding) with probability

$$\cos^4\frac{\theta}{2} + \sin^4\frac{\theta}{2} = 1 - 2\sin^2\frac{\theta}{2}\cos^2\frac{\theta}{2} = 1 - \frac{\sin^2\theta}{2} \quad (16)$$

and a 1 (incorrect decoding) with probability  $\frac{1}{2} \sin^2 \theta$ . Consequently, the eavesdropping detection probability per qubit varies between 0 (realized when  $\theta = 0$ ) and 1/2 (achieved for  $\theta = \pi/2$ ). In other words, Eve remains undetected when the encoding basis coincides with the normal computational basis; on the other hand, there is a 50% probability of detecting the actions of Eve if the encoding basis is "maximally non-orthogonal" to the normal computational basis (like the Hadamard basis  $\{H|0\rangle, H|1\rangle\}$ , for example). Therefore, on average, the detection probability per qubit tested is given by

$$\frac{\frac{1}{2} \int_0^\pi \sin^2 \theta d\theta}{\pi} = \frac{\int_0^{\frac{\pi}{2}} \sin^2 \theta d\theta}{\pi} = \frac{1}{4} \tag{17}$$

As in the more complex scenario discussed before, this probability can be brought as close to 1 as desired by increasing the number of qubits tested for eavesdropping (the signature string).

Another interesting question in this analysis is: How much information from the transmitted message can Eve gain, assuming that she remains undetected? The condition to remain undetected is essential for Eve. Otherwise, Alice will not disclose to Bob the rank (correct position) of each qubit in the message and consequently, the information gain for Eve is null, even if she has correctly decoded each qubit.

A measure of the information gain is  $1 - H_{bin}(p_{Eve}^0)$ , where  $H_{bin}(p_{Eve}^0)$  is the binary entropy associated with the probability of seeing a 0 when measuring a qubit that encodes a 0. We can express this information gain as a function of the parameter  $\theta$  as follows:

$$\begin{aligned} 1 - H_{bin}(p_{Eve}^0) &= 1 + p_{Eve}^0 \log p_{Eve}^0 + (1 - p_{Eve}^0) \log (1 - p_{Eve}^0) \\ &= 1 + \cos^2 \frac{\theta}{2} \log \cos^2 \frac{\theta}{2} + \sin^2 \frac{\theta}{2} \log \sin^2 \frac{\theta}{2} \\ &= 1 + 2 \cos^2 \frac{\theta}{2} \log \cos \frac{\theta}{2} + 2 \sin^2 \frac{\theta}{2} \log \sin \frac{\theta}{2} \end{aligned} \tag{18}$$

The graph of this function is depicted in Fig. 4. When Eve performs a direct measurement in the normal computational basis on a qubit encoding a 0, she can be certain of the observed value for an encoding angle  $\theta$  of 0 or  $\pi$ . On the other extremity, when  $\theta = \frac{\pi}{2}$ , the measurement provides no information gain whatsoever. From Eve's point of view, the message bit could still be a 0 or a 1, with equal probability. On average, the information gain is given by

$$\frac{\int_0^\pi (1 + 2 \cos^2 \frac{\theta}{2} \log \cos \frac{\theta}{2} + 2 \sin^2 \frac{\theta}{2} \log \sin \frac{\theta}{2}) d\theta}{\pi} \approx 0.44 \tag{19}$$

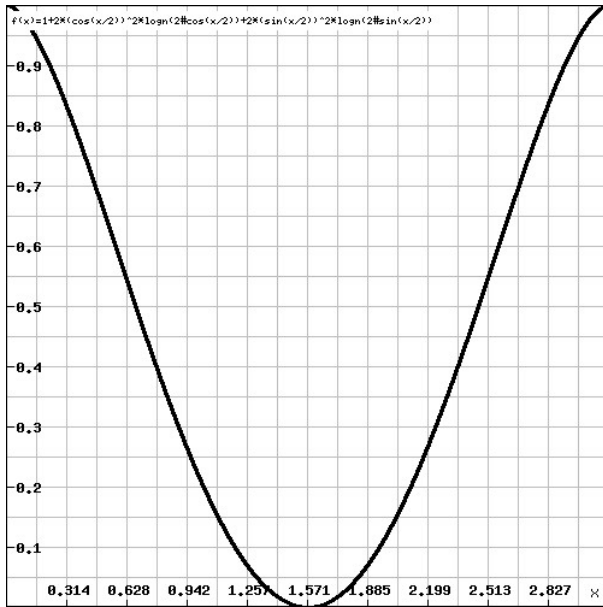


Fig. 4. Information gain as a function of the encoding angle  $\theta$

## 5 Variations

The role of the signature string in this protocol is to ensure (with a certain probability, which can be made arbitrarily large) its security or, in other words, the secrecy of the communication between Alice and Bob. To this end, the bits forming the signature are treated in exactly the same way as the bits composing the actual message: they are scrambled together and encoded according to a chosen basis  $(\theta, \varphi)$ . Consequently, when attempting an eavesdropping, Eve has no knowledge whatsoever if the bit she tampers with is part of the signature or part of the message. Ideally, she would like to eavesdrop only on the message bitstring in order to avoid detection and maximize her knowledge on the message transmitted. Unfortunately for her, the information gain is directly proportional to the probability of being detected, so Eve has to think twice before deciding to eavesdrop on a particular qubit.

This property can be used to slightly simplify the protocol such that the bits in the signature string are not encoded (or equivalently, they are encoded in the normal computational basis: 0 becomes  $|0\rangle$  and 1 becomes  $|1\rangle$ ). Now Eve can completely avoid detection by choosing to measure the bypassing qubits directly in the standard computational basis. In this way, the quantum states of the signature qubits will not be altered, but the state of any message qubit will be projected to one of the two eigenvectors of the measurement basis:  $|0\rangle$  or  $|1\rangle$ .

Consequently, Alice may pick as encoding basis for the message bitstring the Hadamard basis  $(\frac{\pi}{2}, 0)$ , which will maximize Eve’s uncertainty over each

measurement she performs on the message qubits. Effectively, the outcome of each such measurement has a 50% probability of being correct, which is not better than tossing a fair coin. Therefore, we can confidently say that Eve has zero knowledge about the true value encoded in such a qubit, or equivalently, the binary entropy of such a qubit is 1.

This variant of the protocol, in which the message bits are encoded using the Hadamard basis while the bits in the signature are encoded in the normal computational basis, is reminiscent of BB84 [2] with its two encoding bases: horizontal/vertical and diagonal that are randomly applied by Alice. Here, as there, it is the "maximum non-orthogonality" of the two bases that keeps Eve in the dark, but this protocol has an important advantage: it can be used to directly encrypt any message without the need to establish a secret key first. Still, the classical channel needs to be authenticated, which is usually done with a small secret key, but it was shown that this requirement is too strong and all that is actually needed is protected public information [7].

Since the price for avoiding detection is total uncertainty about the transmitted message, Eve is forced to measure at least some of the qubits in the Hadamard basis, thus exposing herself to detection. The choice for Eve is a difficult one: either try as much as possible to remain hidden, but then she faces the prospect of gaining little information (if any at all) about the content of the message, or aiming at decrypting as much as possible from the transmitted message, which increases the risk of being caught. And in case of detection, no information is gained (zero knowledge), because Alice will no longer reveal the proper order of the bits in the message.

If the first variation discussed is one in which the complexity of the protocol is decreased, the second one involves an increase in complexity with the purpose of also decreasing the probability of the worst case. In the original protocol, the worst case happens when Eve gets so lucky that she guesses precisely the encoding basis  $(\theta, \varphi)$  used by Alice. We can think of a variation in which Alice randomly chooses a pair  $(\theta, \varphi)$  for every single bit transmitted. This will not change the average-case analysis, but will definitely make the worst-case much less probable, since Eve will have to get lucky  $N$  times now, where  $N$  is the total number of bits transmitted (message and signature together).

## 6 Conclusion

In this paper, we have developed a novel quantum protocol that allows two communicating parties to exchange classical messages directly, without the need to establish a secret key prior to the communication. Quantum mechanical properties have been used before in cryptographic protocols, but only for key distribution purposes, or more precisely for key enhancement, if authentication is achieved through a small secret key already distributed to the parties involved. Our protocol is entanglement-free and uses only unary quantum transformations, which means that the computational power assumed is less than that of a universal quantum computer. Consequently, communicating securely through

public channels can be performed simple, fast and efficient if we resort to quantum mechanics to directly encrypt the transmitted message into a sequence of qubits.

Two important ideas made this result possible: the use of a quantum memory and bringing the rank (or position) of a bit in the message bitstring into play. The use of a quantum memory is essential in order to make "informed" measurements in the second phase of the protocol, after all the qubits have been received. Yet, storing qubits is rarely contemplated (if ever) in quantum protocols, perhaps due to their fragility and ephemeral nature. Nevertheless, experimental quantum physicists are making good progress towards making quantum memories a practical reality.

Scrambling the qubits encoding the message, on the other hand, guarantees that no knowledge whatsoever about the content of the message is gained by a potential eavesdropper, even in the highly unlikely eventuality of correctly decoding the individual bits in the message. It is our belief that the synergy of these two ideas working together may open the door for a whole new class of cryptographic protocols with superior characteristics.

## References

1. Bennett, C.H.: Quantum cryptography using any two nonorthogonal states. *Physical Review Letters* 68(21), 3121–3124 (1992)
2. Bennett, C.H., Brassard, G.: Quantum cryptography: Public key distribution and coin tossing. In: *Proceedings of IEEE International Conference on Computers, Systems and Signal Processing*, Bangalore, India, pp. 175–179. IEEE, New York (1984)
3. Bennett, C.H., Brassard, G., Mermin, N.D.: Quantum cryptography without Bell's theorem. *Physical Review Letters* 68(5), 557–559 (1992)
4. Ekert, A.: Quantum cryptography based on Bell's theorem. *Physical Review Letters* 67, 661–663 (1991)
5. Gisin, N., et al.: Quantum storage of photonic entanglement in a crystal. *Nature* 469, 508–511 (2011)
6. Lukin, M., et al.: Room-Temperature Quantum Bit Memory Exceeding One Second. *Science* 336, 1283–1286 (2012)
7. Nagy, N., Akl, S.G.: Authenticated quantum key distribution without classical communication. *Parallel Processing Letters, Special Issue on Unconventional Computational Problems* 17(3), 323–335 (2007)
8. Rivest, R.L., Shamir, A., Adleman, L.M.: A method of obtaining digital signatures and public-key cryptosystems. *Communications of the ACM* 21(2), 120–126 (1978)
9. Steger, M., Saeedi, K., Thewalt, M.L.W., Morton, J.J.L., Riemann, H., Abrosimov, N.V., Becker, P., Pohl, H.J.: Quantum Information Storage for over 180s Using Donor Spins in a Si<sup>28</sup> "Semiconductor Vacuum". *Science* 336(6086), 1280–1283 (2012)
10. Tittel, W., et al.: Broadband waveguide quantum memory for entangled photons. *Nature* 469, 512–515 (2011)

# Quantum Secret Communication without an Encryption Key

Marius Nagy and Naya Nagy

College of Computer Engineering and Science  
Prince Mohammad Bin Fahd University, Al Khobar, Saudi Arabia  
{mnagy, nnagy}@pmu.edu.sa

**Abstract.** Quantum cryptographic methods increase security over classical methods. To date, quantum algorithms aim to distribute a secret key to be used afterwards to encrypt messages. The method described in this paper does not use an encryption key at all. An array of qubits is transmitted from the source to the destination with the message encoded in the phase of the qubit. The secrecy of the message derives from the nonclonability principle. Our algorithm relies on the common assumption that public information can be authenticated. The algorithm shows an increased detection rate per qubit, 33%, which is higher than the one commonly used in literature, namely 25%.

**Keywords:** Quantum Key Distribution, Quantum Cryptography, Intruder Detection.

## 1 Introduction

Quantum cryptography has been mainly concerned with quantum key distribution. The two communicating parties, Alice and Bob, undergo a protocol to distribute a secret key. Alice and Bob aim to reach a consensus on the value of a secret key. This key is to be used **later** to encrypt/decrypt a message. Actually, a large body of literature considers the quantum key distribution problem to be in fact a key enhancement [5]. Nevertheless, the opposite opinion, which says that true key distribution can be achieved with quantum means only, also has its adepts [3]. Key enhancement means that Alice and Bob share already a small secret key, possibly obtained via a classical protocol, and then develop a large secret key. Key distribution starts from public information only and develops a secret key during the protocol.

In this paper, we distance ourselves from the very idea of using a **key** for encryption. We develop a protocol that transmits a message secretly by scrambling the order of the bits rather than explicitly encrypting the message with a key. The scrambled message is transmitted via a quantum channel and therefore consists of quantum bits (qubits) rather than binary bits.

Our protocol comes with all the advantages of quantum cryptography. An intruder, Eve, listening to the message being transmitted, destroys the superposition of the qubits and thus can gain knowledge about it only with a low

probability. Also, the intruder is detected by Alice and Bob with an arbitrarily high probability.

Additionally, our protocol is equivalent to a one-time-pad [6] protocol. As we use no key, information about the scrambling of the message is of the same length as the message itself. Eavesdropping one message provides no gain to the intruder for any subsequent messages.

Previous quantum key distribution protocols [1] [2] have a detection rate of 25% per checked qubit. We develop an encoding strategy in three complementary basis that improves the detection rate per qubit to 33%.

The rest of the paper is organized as follows. Section 2 presents the keyless protocol that securely transmits a message from a source to a destination. It also analyzes the protocol's protection to the intruder's actions. The analysis is formalized to measure the intruder's gain of knowledge for different levels of attack. Section 3 describes an improvement on the detection rate of the intruder by using an encoding in three complementary bases. Section 4 concludes the paper.

## 2 Keyless Quantum Message Transmission

Using Dirac's notation, a qubit is  $q = \alpha|0\rangle + \beta|1\rangle$ .  $\alpha$  and  $\beta$  are complex numbers. Thus,  $|\alpha|^2$  is the probability of the qubit to collapse to 0, and  $|\beta|^2$  to 1. Qubits are said to be in a balanced superposition if the qubit has an equal chance 50% to collapse to 0 or 1. Quantum protocols use a small set of common gates. Three such gates are used in our protocols: the controlled-NOT (CNOT) gate, the Hadamard gate, and the phase-shift gate [4]. All these gates have a control qubit. If the control qubit is  $|1\rangle$ , the primary qubit is transformed according to the gate's definition. If the control qubit is  $|0\rangle$ , the primary qubit passes the gate undisturbed.

In this section, we describe in detail the inner workings of a protocol that allows two parties, Alice and Bob, to communicate secretly over an insecure, public quantum channel. The protocol relies on the fact that a quantum channel cannot be eavesdropped on without disturbing the quantum information transmitted over the channel. In addition to the quantum channel, the quantum protocol also requires an authenticated channel for classical communication and a quantum memory (i.e. the ability to store the states of a certain number of qubits for a certain amount of time). The main steps of the protocol are:

### Phase I: Communication over the Quantum Channel

- Step 1:** Alice concatenates the two binary strings, one representing the message she intends to send over to Bob and the other representing the signature bitstring that will be used for eavesdropping.
- Step 2:** For each bit in the concatenated sequence, Alice uses one of the two bases, or alphabets (chosen randomly) to encode the value of the respective bit in the quantum state of the resulting qubit.



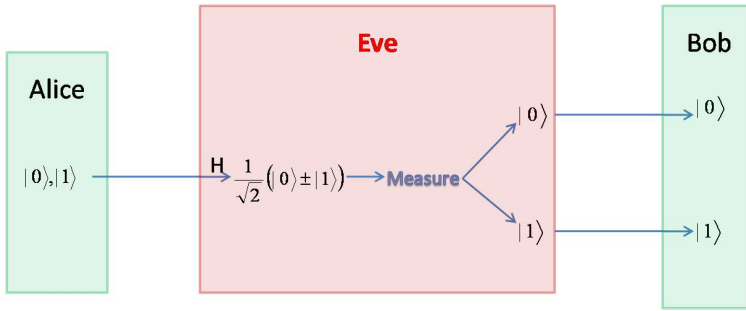
- Step 3:** Alice scrambles the order of the qubits forming the quantum encrypted block obtained in step 2, by choosing an arbitrary permutation of the qubits and then sends them over to Bob through the insecure, public quantum channel.
- Step 4:** Bob applies the necessary procedures to safely store the qubits received from Alice until the second phase of the protocol, when he will gain knowledge about each qubit's encoding basis and position in the original qubit sequence. The position, or index of the qubit in the original sequence is called the qubit's rank.

## Phase II: Communication over the Classical Channel

- Step 1:** Alice discloses to Bob which of the qubits transmitted are part of the signature string and the encoding base of each.
- Step 2:** Following Alice's instructions, Bob reconstructs the signature bitstring.
- Step 3:** Alice and Bob proceed to verify, bit by bit, whether the signature bitstring was untampered with, during the transmission.
- Step 4:** If the discrepancy between Alice and Bob is discovered in the values of the signature bits, the presence of an eavesdropper is inferred and the protocol is abandoned. Otherwise, Alice informs Bob about the correct position (rank) of each qubit in the original message and the encoding alphabet employed to obtain each qubit.
- Step 5:** Bob decodes and re-arranges the qubits he still has in storage in order to obtain the plain message sent to him by Alice.

Having presented the structure of the protocol, a few clarifications and an analysis of it are perhaps appropriate at this point. Generally, the length of the signature bitstring reflects the intended level of security for the transmitted message. As the analysis below clearly shows, a longer signature bitstring results in higher chances of detecting a potential eavesdropper. Consequently, the signature length can be varied according to the importance of the message.

The protocol above is described in general terms, abstracted away from any particular physical realizations for a qubit. Moreover, any two alphabets, i.e. encoding bases, can be used, as long as they are complementary. Complementary bases means that they correspond to conjugate quantum variables. In this situation, trying to measure (decode) a qubit using the other basis, and not the one used for encoding, will maximize the uncertainty over the value of the corresponding bit: equal chances to obtain 0 or 1. From a mathematical point of view, the simplest example to achieve complementarity would probably be the use of the regular computational basis  $\{|0\rangle, |1\rangle\}$  together with the "Hadamard basis"  $\{H|0\rangle = \frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}}, H|1\rangle = \frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}}\}$ . We note in passing that the BB84 protocol [1], which uses photon polarization as qubit embodiment, achieves com-



**Fig. 1.** Opaque eavesdropping. Eve wrongly measures in the Hadamard basis a qubit sent by Alice in the computational basis.

plementarity by choosing randomly between rectilinear polarization  $\{| \rightarrow \rangle, | \uparrow \rangle\}$  and diagonal polarization  $\{| \nearrow \rangle, | \nwarrow \rangle\}$  as the two possible encoding bases. In general, the precise meaning or interpretation of a certain basis depends entirely on the physical realization chosen for the qubit. To keep our discussion as general as possible, while still referring to a concrete pair of complementary bases, we assume henceforth that the two encoding alphabets are the computational basis and the Hadamard basis, as specified above. This basically means that Alice will create a  $|0\rangle$  qubit for each 0 bit in the message and a  $|1\rangle$  qubit for each 1 bit in the message, with a random choice to apply a Hadamard gate on the resulting qubit. What can Eve, the prototypical eavesdropper do, in order to elicit as much information as possible about the transmitted message, while the qubits are in transit from Alice to Bob? The two main possible eavesdropping strategies are discussed next.

## 2.1 Opaque Eavesdropping

Opaque eavesdropping refers to Eve's attempt to gain knowledge about the transmitted message by measuring each qubit passing through the quantum channel in one of the two possible bases. Eve knows the two bases that she has used: computational and Hadamard. Yet, for any specific qubit, Eve does not know the basis used, as Alice chooses the basis randomly. If Eve is lucky and chooses the same basis, she will be able to read the binary value of the qubit and will leave no trace of her interference. Nevertheless, if Eve chooses the wrong basis, she gains no knowledge about the binary value of the qubit, and also may disturb the correct measurement for Bob. There are two cases with similar results. First, Alice may send the qubit simply in the computational basis, see fig. 1. If Eve mistakenly applies a Hadamard gate prior to her own measurement, she will get either 0 or 1 with equal probability, regardless of Alice's original value. Therefore, Bob may measure the wrong value with a 50% chance. If this is a qubit that Alice and Bob check, again they have a 50% chance to catch Eve. Secondly, Alice may send a qubit in the Hadamard basis. If Eve mistakenly measures the qubit directly she again produces a qubit on which she may be caught with a chance

of 50%. Therefore, on each qubit that Eve wrongly disturbs, she is caught 50% of the times. As she is disturbing half the qubits on average, Eve is caught with a probability of 25% on each qubit she chooses to observe. Or else, on each qubit that Eve decides to observe and Bob decides to check, Eve remains undetected with a probability of  $75\% = \frac{3}{4}$ .

Suppose, there are  $n$  qubits in the signature string. They are observed by Eve and checked by Bob. Eve remains undetected with a probability of  $(\frac{3}{4})^n$ . Therefore Bob's detection rate over  $n$  qubits is given by the formula  $rate = 1 - (\frac{3}{4})^n$ .

Nevertheless, if Eve gets lucky enough to remain undetected, then she will gain access to the rank and encoding basis of each bit in the message. This means that she can put the bits in the correct order, but she can only be certain about their value for half of them, the ones for which she correctly guessed the encoding basis. For example, if Eve listens to  $n$  qubits, she is certain of the value of  $\frac{n}{2}$  qubits. Thus, her *information gain* is  $50\% = \frac{1}{2}$ .

Note that the probability for Eve to remain undetected may be very low; for example, if the signature string is 25 bits long, Eve remains undetected with a probability of about 0.075%.

## 2.2 Translucent Eavesdropping

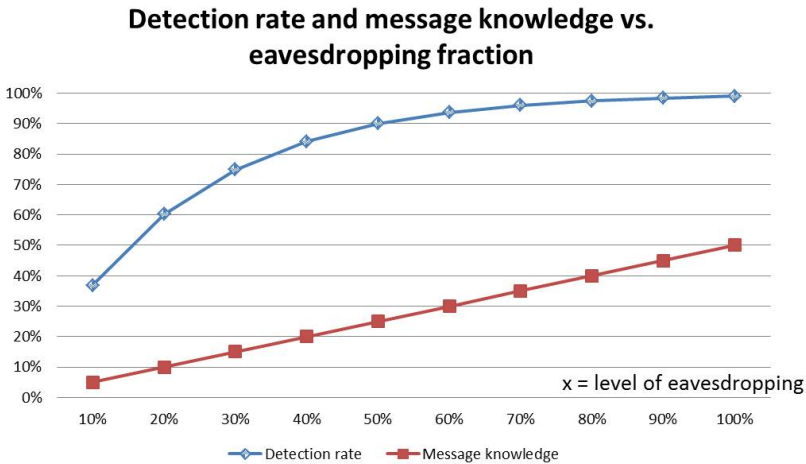
Alternatively, Eve could try a more insidious eavesdropping strategy, avoiding a direct measurement on the qubits in transit through the quantum channel. This can be achieved by making a copy of each qubit or entangling each qubit to one of her own, before sending the original further on to Bob. Since the two encoding bases are complementary, no quantum circuit exists that can accurately duplicate all four base vectors (no-cloning theorem). For example, the Controlled-NOT (CNOT) gate acts as a cloning gate for qubits encoded in the computational basis, but creates an entangled pair  $\frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$  whenever we push a quantum state like  $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$  through it. Consequently, each qubit originally encoded by Alice in the Hadamard basis, will arrive at Bob entangled with a corresponding qubit in Eve's possession. Now when Bob applies a Hadamard gate on his half of the entanglement, in order to decode the qubit, he effectively transforms the state of the Bob-Eve ensemble as follows:  $H \otimes I \left( \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle \right) = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$ , and  $H \otimes I \left( \frac{1}{\sqrt{2}}|00\rangle - \frac{1}{\sqrt{2}}|11\rangle \right) = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle + |11\rangle)$ .

When any of the two quantum states above is measured by Bob in the normal computational basis, the entanglement will collapse to one of the four basis vectors  $\{ |00\rangle, |01\rangle, |10\rangle, |11\rangle \}$  and Bob will have a 50% chance to obtain the correct bit value, the one originally encoded by Alice. Consequently, the detection rate for translucent eavesdropping is the same as the one derived for opaque eavesdropping.

### 2.3 Lower Levels of Eavesdropping

The above analysis for eavesdropping consequences is based on the assumption that Eve tampers with all qubits transmitted through the quantum channel. Here, tampering with a qubit means either measuring or trying to clone it. If Eve is caught, she gains no knowledge whatsoever about the content of the message. This happens because whenever Eve is caught in Step 4 of Phase II of the protocol, see section 2, the protocol is abandoned. Alice does not reveal the correct order of the qubits and the scrambled message is meaningless both to Eve and Bob.

Consequently, Eve could settle for a more discrete strategy, according to the plan that partial information is better than no information at all. If Eve decides to eavesdrop on a fraction  $x$  for the qubits in the quantum encrypted block transmitted, then the detection rate varies with  $x$  and with the signature length  $n$  as follows:  $rate = 1 - \left(\frac{3}{4}\right)^{x \cdot n}$ , where  $0 \leq x \leq 1$  and  $n$  is the length of the signature, for example  $n = 16$  bits long.



**Fig. 2.** The graph shows the detection rate together with Eve’s information gain. The  $Ox$  axis represents  $x$ , the percentage of the signature read by Eve. The  $Oy$  axis shows both the detection rate and the information gain.

In the eventuality that she remains undetected, the percentage of the message that Eve is certain she has correctly decoded is 50%. Thus the *information gain* on a fraction  $x$  is  $\frac{x}{2}$ . A graph depicting the variation of the detection rate and information gain for various levels of eavesdropping is presented in fig. 2. The graph assumes a constant signature length of 16 bits. A longer signature will, of course, push the detection rates asymptotically closer to the 100% limit.

From Eve’s point of view, probably the most pertinent question is: *What is the optimal level of eavesdropping such that the probability of escaping detection and the*

knowledge gained about the message are both maximized? In order to answer this question, we need to find the maximum of a benefit function that quantifies both these quantities. A suitable function is  $f_{benefit} : [0, 1] \rightarrow [0, 1]$ ,  $f_{benefit}(x) = \frac{x}{2} \left(\frac{3}{4}\right)^{x \cdot n}$ .

This function was obtained by multiplying the two quantities, probability of escaping detection and the fraction of the message correctly decoded, normalized to the interval  $[0, 1]$ . As it can be seen from fig. 3, this function reaches its maximum for a level of eavesdropping of about 22%, if the signature string consists of 16 bits. This maximum drops to 14% for a 24-bit signature and to around 11% for a 32-bit signature. These data suggest that the best strategy for Eve is to decrease the level of eavesdropping as the size of the signature increases. However, the length of the signature string is disclosed only during the second phase of the protocol, so Eve cannot use this information in planning her eavesdropping strategy.

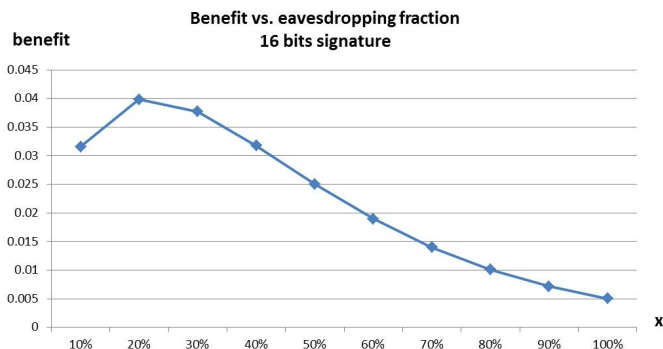


Fig. 3. The benefit of eavesdropping versus the detection rate

### 3 Encoding in Three Bases

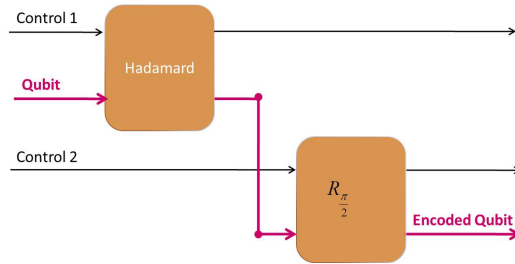
We have discussed an algorithm that reveals the presence of Eve whenever the signature test fails. For each bit of the signature, Eve can be detected with a probability of 25%. This detection rate per qubit is common to all classical key distribution protocols [1] [2]. We hereby propose an encoding scheme that improves the detection rate per qubit to 33%. The improved detection rate comes from encoding each qubit in three complementary bases. While this may seem to increase the complexity in manipulating each qubit, yet the gates used for encoding are common and simple.

The three bases used for encoding are the computational basis, the Hadamard basis, and the phase-shift- Hadamard basis. The phase-shift- Hadamard basis has two gates applied to a qubit: a Hadamard gate and then a  $R_{\frac{\pi}{2}}$  rotation.

When Alice wants to send a binary digit 0 or 1, she first prepares a qubit in the computational basis  $|0\rangle$  or  $|1\rangle$ . Then Alice chooses randomly one of the three bases to encode her qubit:

1. The computational basis  $|0\rangle$  and  $|1\rangle$ .
2. The Hadamard basis,  $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  for 0 and  $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$  for 1.
3. The  $R_{\frac{\pi}{2}}$ -Hadamard basis,  $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$  for 0 and  $\frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$  for 1.

If Alice chooses the computational basis, she simply sends the qubit to Bob. If Alice chooses the Hadamard basis, then Alice applies a Hadamard gate first and then sends the transformed qubit to Bob. If Alice chooses the  $R_{\frac{\pi}{2}}$ -Hadamard basis, Alice applies a Hadamard gate then a  $\frac{\pi}{2}$  phase shift gate, and then sends the doubly transformed qubit to Bob.



**Fig. 4.** Encoding of a qubit in three orthogonal bases. The random values of the control qubits 1 and 2 define the actual encoding basis.

As Alice has three options, the choice can be made by two control bits that are set on arbitrary values. Fig. 4 shows the quantum circuit that Alice uses to encode each qubit. The table below shows the encoding basis as given by the values of the two control bits.

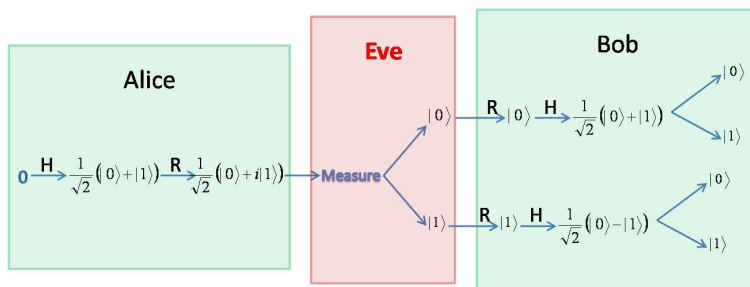
Control 1	Control 2	Encoding Basis
0	0	computational basis
0	1	not used
1	0	Hadamard basis
1	1	phase-shift Hadamard basis

According to the protocol, when Bob receives a qubit from Alice, he waits to be informed on the classical channel what encoding basis was used. Then he applies the necessary gates in reverse order: the phase-shift gate first and then the Hadamard gate.

### 3.1 What Eve Can Do

The eavesdropper can be supposed to know the mechanism of encryption, while not knowing the values of the random control bits.

In opaque eavesdropping, Eve will try to measure the qubit intercepted from Alice and then will further transmit either the measured qubit or a qubit of her



**Fig. 5.** Alice encodes her qubit in the phase-shift-Hadamard basis. Eve guesses the computational basis. Bob catches Eve with a 50% chance.

choice to Bob. Eve guesses one of the three encoding bases and treats the qubit intercepted from Alice accordingly.

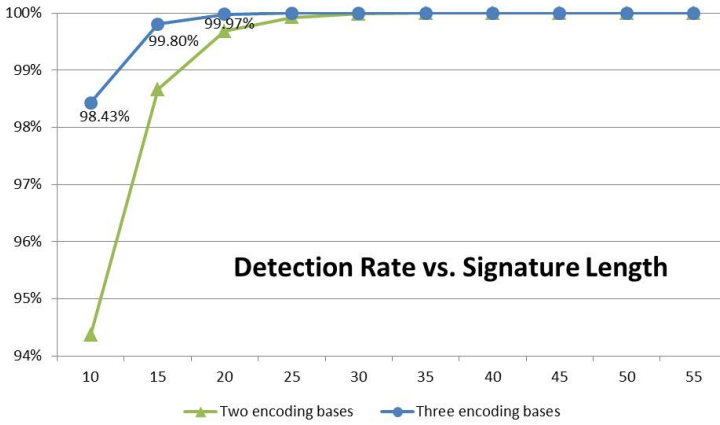
Suppose Eve tries the computational basis. If Alice’s qubit is encoded in the computational basis, Eve reads the correct value and remains undetected. If Alice’s qubit is encoded in the Hadamard basis, Eve wrongly pushes Alice’s qubit through a Hadamard gate and will be detected by Bob in 50% of the cases. This situation was represented in fig. 1. If Alice’s qubit was encoded in the phase-shift -Hadamard basis and Eve measures the qubit in the computational basis, Eve destroys the balanced superposition. As in the previous case, Bob can catch Eve with a 50% chance. Fig. 5 shows an example of Alice encoding a binary 0 in the phase-shift-Hadamard basis. Bob, by applying the same steps that Alice did in reverse order will retrieve the initial 0 only 50% of the times. As Alice encodes a qubit randomly in one of the three bases, and Eve reads the stolen qubit in the computational basis, Eve will be caught in two situations with a chance of 50%. This yields an overall probability of  $\frac{1}{3}(\frac{1}{2} + \frac{1}{2}) = 33\%$ . This chance is considerably higher than 25% offered by two bases encoding.

We supposed that Eve decides to measure the intercepted qubit in the computational basis. If Eve chooses to measure in any other of the three bases, a similar result can be deduced. The detection probability is 33% no matter what basis Eve chooses.

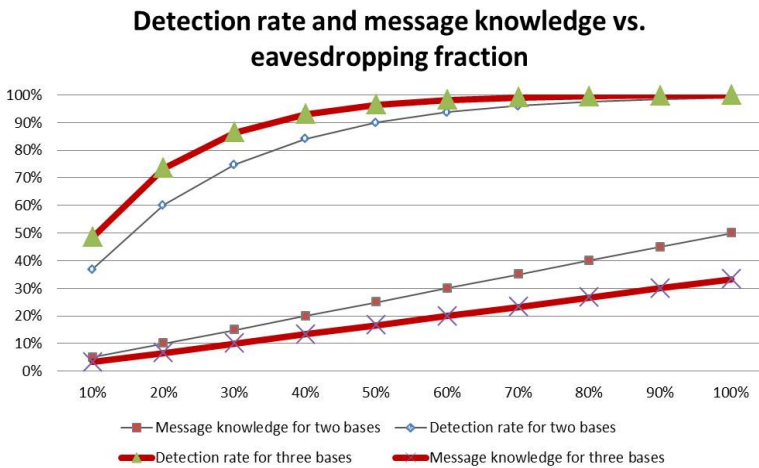
If eavesdropping is tested on a larger signature, the detection rate increases sharply with the length of the signature  $n$ :  $rate = 1 - (\frac{2}{3})^n$ .

Fig. 6 shows a comparison on the detection rate for the case of two encoding and three encoding bases respectively. The graph shows that for short signatures, the detection rate for three encoding bases is measurably larger, whereas signatures large than 25 qubits do not benefit from three encoding bases.

**Lower Levels of Eavesdropping.** Let us study the optimal level of eavesdropping on the three bases encoding scheme. Under the assumption that Eve is not caught, Eve gains the value of the qubits that she has luckily measured in the same basis as Bob. As there are three possible bases, Eve reads correctly  $\frac{1}{3}$  of the qubits she intercepts.



**Fig. 6.** The graph shows the detection rate versus the signature length for three encoding bases. The  $Ox$  axis represents the length of the signature string. The  $Oy$  axis shows the probability for Eve to be detected.

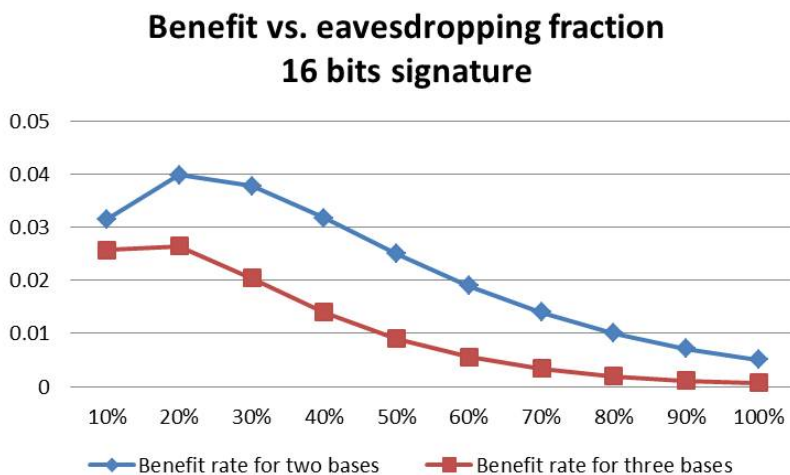


**Fig. 7.** The graph shows the detection rate together with Eve’s information gain. The  $Ox$  axis represents the percentage of the signature read by Eve. The  $Oy$  axis shows both the detection rate and the information gain.



Suppose Eve does not listen to the entire qubit block, but eavesdrops a fraction  $x$ . Therefore, she will disturb a fraction  $x$  of the signature of length  $n$ . The detection rate varies with  $x$  according to the following formula  $rate = 1 - (\frac{2}{3})^{x \cdot n}$ .

Also,  $x$  affects the information gain, which will be the fraction  $\frac{x}{3}$  of the message. Fig. 7 represents both the detection rate and the information gain for the three bases encoding scheme, computed on a signature of 16 bits. The graphs for a two bases encoding are also shown for comparison in the figure, with a thin line. It can be seen that the three base protocol improves over the two base protocol, both in terms of detection rate as well as information gain.



**Fig. 8.** The benefit of eavesdropping versus the detection rate

In section 2.3, we defined a benefit function that Eve uses to find the optimal level of eavesdropping. For the three bases encoding, the function becomes  $f_{benefit} : [0, 1] \rightarrow [0, 1]$ ,  $f_{benefit}(x) = \frac{x}{3} (\frac{2}{3})^{x \cdot n}$ .

Fig. 8 shows the graph of this function juxtaposed with the graph for the two bases encoding defined in section 2.3. By comparison, we see that the optimal level of eavesdropping is approximately the same, about 22%. Nevertheless, for a three bases encoding scheme the benefit is considerably lower.

## 4 Conclusion

We have shown that secret communication does not need an encryption key. The previous section contains a protocol that transmits a secret message without encoding the message with a key. The secrecy of the message ensues from randomly scrambling the order of the bits in the message. As the bits are sent in random

order, the scrambled message does not reveal anything about the content of the message. The correct order of the qubits is revealed publicly after the absence of an intruder is checked.

The protocol benefits from the capability of detecting an intruder. This is a major characteristic of all quantum key distribution protocols. The intruder, Eve, leaves an unmistakable mark on the qubits she read: she changes the intended value of the qubit with a certain probability. Our paper has an improved detection rate of Eve from 25% to 33% per intercepted qubit. This is achieved by using three orthogonal encoding bases. Eve's presence is searched on a signature, as in all other protocols.

Our paper gives an extensive analysis on what Eve can do: opaque and translucent eavesdropping, and also low levels of eavesdropping. It studies the advantages of Eve and the maximum benefit Eve can get from a certain signature length.

## References

1. Bennett, C.H., Brassard, G.: Quantum cryptography: Public key distribution and coin tossing. In: Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India, December 1984, pp. 175–179. IEEE, New York (1984)
2. Bennett, C.H.: Quantum cryptography using any two nonorthogonal states. *Physical Review Letters* 68(21), 3121–3124 (1992)
3. Nagy, N., Nagy, M., Akl, S.G.: Key distribution versus key enhancement in quantum cryptography. *Parallel Processing Letters* 20(03), 239–250 (2010)
4. Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information*. Cambridge University Press (2000)
5. Lomonaco Jr., S.J.: A Talk on Quantum Cryptography or How Alice Outwits Eve. In: Proceedings of Symposia in Applied Mathematics, Washington, DC, vol. 58, pp. 237–264 (January 2002)
6. Shannon, C.: Communication theory of secrecy systems. *Bell System Technical Journal* 28(4), 656–715 (1949)

# Securely Computing the Three-Input Majority Function with Eight Cards

Takuya Nishida<sup>1</sup>, Takaaki Mizuki<sup>2</sup>, and Hideaki Sone<sup>2</sup>

<sup>1</sup> Sone-Mizuki Lab., Graduate School of Information Sciences, Tohoku University,  
6-3 Aramaki-Aza-Aoba, Aoba, Sendai 980-8578, Japan  
`nisida@s.tohoku.ac.jp`

<sup>2</sup> Cyberscience Center, Tohoku University,  
6-3 Aramaki-Aza-Aoba, Aoba, Sendai 980-8578, Japan  
`tm-paper+tpnc2013@g-mail.tohoku-university.jp`

**Abstract.** Assume that Alice, Bob and Carol, each of whom privately holds a one-bit input, want to learn the value of the majority function of their inputs without revealing more of their own secret inputs than is necessary. In this paper, we show that such a secure majority computation can be done with a deck of real cards; specifically, the three players can learn only the majority of their inputs using eight physical cards—four black cards and four red cards—with identical backs.

**Keywords:** Card-based protocols, Card games, Cryptography without computers, Recreational cryptography, Secure computations.

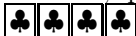


## 1 Introduction

Assume that there are three players, Alice, Bob and Carol, who privately hold one-bit inputs  $a \in \{0, 1\}$ ,  $b \in \{0, 1\}$  and  $c \in \{0, 1\}$ , respectively. The private inputs could be, for example, their YES/NO answers (held in mind) regarding their support for some new plan. In such a case, majority voting is often conducted to find the majority of their answers, and sometimes it is not preferable to reveal all of the individual inputs. Thus, we assume that Alice, Bob and Carol all want to learn the value of the *majority function*  $\text{maj}(a, b, c)$  of their inputs  $a, b, c \in \{0, 1\}$  without revealing more of their own private inputs than is necessary, i.e., they wish to know only the value of

$$\text{maj}(a, b, c) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } a + b + c \geq 2; \\ 0 & \text{if } a + b + c \leq 1. \end{cases}$$

Note that

$$\text{maj}(a, b, c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a).$$





In this paper, we show that such a *secure majority computation* can be conducted using a deck of ‘real’ cards; specifically, it can be done using eight physical cards—four black cards  and four red cards —with identical backs (). As shown below, our secure majority computation, which is a

kind of cryptographic task, relies on simple properties of physical cards; we do not need a conventional computer or communication network system.



This paper starts with a review of *card-based cryptographic protocols*.

### 1.1 Card-Based Cryptographic Protocols

Card-based cryptographic protocols enable us to perform secure computations. The mainstream research on secure computations (e.g. [5,13]) initiated by the seminal work of Yao [15] usually aims to implement protocols on computers communicating with each other via network systems. In contrast, card-based protocols require only a small deck of cards, which is portable, handy and inexpensive, and furthermore, needs no electricity. Also, it is easy for even non-specialists to understand why card-based protocols can compute functions properly while maintaining secrecy. Considering the fact that, even in the digital era, elections are often conducted with physical paper ballots, we believe that physical implementations of secure computations are still quite important (e.g. [1,6,10,11]).

We first mention the properties of the cards appearing in this paper. All cards of the same type ( or ) are assumed to be indistinguishable from one another. We use  to denote a card lying face down. We also assume that the backs  of all cards are identical. To deal with Boolean values, we use the following encoding:

$$\begin{matrix} \clubsuit & \heartsuit \\ \boxed{?} & \boxed{?} \end{matrix} = 0, \quad \begin{matrix} \heartsuit & \clubsuit \\ \boxed{?} & \boxed{?} \end{matrix} = 1. \tag{1}$$

Next, we define a “commitment.” Given a bit  $x \in \{0, 1\}$ , a pair of face-down cards  whose value is equal to  $x$  (according to the encoding rule (1) above) is called a *commitment to  $x$*  and is expressed as

$$\underbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}_x.$$

Note that swapping the two cards constituting a commitment to a bit  $x$  results in a commitment to negation  $\bar{x}$ :

$$\underbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}_x \rightarrow \overbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}^{\neq} \rightarrow \underbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}_{\bar{x}}.$$

Thus, a secure NOT operation is trivial.

Several card-based protocols have been described in the literature for securely computing other operations such as AND, XOR and Adder, as listed in Table 1. There are two types of protocols with regard to output format. The first two protocols in Table 1 produce their outputs (the value of  $a \wedge b$ ) publicly, whereas the remaining nine protocols produce their outputs in a *committed format*, i.e., their output is described as a sequence such as

$$\underbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}_{a \wedge b} \quad \text{and} \quad \underbrace{\begin{matrix} \boxed{?} & \boxed{?} \end{matrix}}_{a \oplus b}$$

following the encoding rule (1).

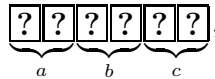
**Table 1.** Known card-based protocols for secure computation

	# of colors	# of cards	Avg. # of trials
◦ <i>Secure AND in a non-committed format</i>			
den Boer [2]	2	5	1
Mizuki-Kumamoto-Sone [7]	2	4	1
◦ <i>Secure AND in a committed format</i>			
Crépeau-Kilian [3]	4	10	6
Niemi-Renvall [11]	2	12	2.5
Stiglic [14]	2	8	2
Mizuki-Sone [8] (§2.2)	2	6	1
◦ <i>Secure XOR in a committed format</i>			
Crépeau-Kilian [3]	4	14	6
Mizuki-Uchiike-Sone [9]	2	10	2
Mizuki-Sone [8]	2	4	1
◦ <i>Secure Half Adder in a committed format</i>			
Mizuki-Asiedu-Sone [6]	2	8	1
◦ <i>Secure Full Adder in a committed format</i>			
Mizuki-Asiedu-Sone [6]	2	10	1

In addition to the protocols listed in Table 1, there are copy protocols, one of which is introduced later, in Section 2.3.

### 1.2 Our Results

Recall our goal: given three commitments



we desire to obtain a commitment



to the majority. As mentioned in the previous subsection, the card-based protocol enables us to perform secure computations of NOT, AND and XOR in a committed format as well as to securely copy commitments. Consequently, it is well known that *any* function can be securely computed using some number of cards. In fact, using the existing protocols (which are described later in Section 2), we can construct a trivial protocol for securely computing the majority function  $\text{maj}(a, b, c)$ , although doing so needs eight additional cards, by using 14 cards in total (including the six cards for the three commitments), as shown in Section 3.

In contrast, this paper shows that such a secure majority computation can be done with only two additional cards, namely eight cards in total (as mentioned before). Our novel protocol is presented in Section 4. This paper concludes in Section 5 with some discussions and open problems.

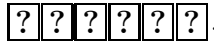
## 2 Known Protocols

In this section, after explaining a shuffling operation called a “random bisection cut,” we introduce an existing AND protocol and an existing copy protocol [8] for our use later (and for the readers to become familiar with card-based protocols).

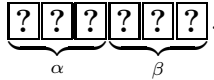
### 2.1 Random Bisection Cuts

A random bisection cut was invented in [8]. We demonstrate it by taking six cards as an example.

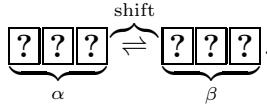
1. Assume that there are six cards as follows:



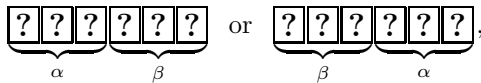
2. Bisect the deck of cards, and let the two sections be  $\alpha$  and  $\beta$ :



3. Shift  $\alpha$  and  $\beta$  randomly:

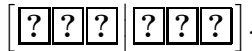


4. After applying such a random shift, the cards are either in their initial state or in a shifted state, as follows:





each of which occurs with probability of exactly  $1/2$ .

This kind of shuffling is referred to as a *random bisection cut* denoted by  $[\cdot | \cdot]$ . Below is the expression of a random bisection cut for six cards:



### 2.2 The Six-Card AND Protocol

Using the random bisection cut, we can construct a six-card AND protocol [8] that securely computes the function  $f(a, b) = a \wedge b$  with a total of six cards: three s and three s.

Before going into the details of the protocol, we define two operations, **get** and **shift**. Given a pair of bits  $(x, y)$ , we define

$$\begin{aligned} \text{get}^0(x, y) &= x; \\ \text{get}^1(x, y) &= y; \\ \text{shift}^0(x, y) &= (x, y); \\ \text{shift}^1(x, y) &= (y, x). \end{aligned}$$

Thus,  $\text{get}^0(x, y)$  returns the first bit,  $\text{get}^1(x, y)$  returns the second bit,  $\text{shift}^0(x, y)$  returns the two bits identically, and  $\text{shift}^1(x, y)$  swaps the two bits.

Note that using the operations above, the AND function can be written as

$$a \wedge b = \text{get}^a(0, b) = \text{get}^a(\text{shift}^0(0, b))$$

because

$$a \wedge b = \begin{cases} 0 & \text{if } a = 0; \\ b & \text{if } a = 1. \end{cases}$$

Furthermore, since

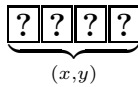
$$\text{get}^a(0, b) = \text{get}^{a \oplus 1}(b, 0) = \text{get}^{a \oplus 1}(\text{shift}^1(0, b)),$$

it holds that

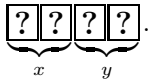
$$a \wedge b = \text{get}^{a \oplus r}(\text{shift}^r(0, b)) \tag{2}$$

for a random bit  $r \in \{0, 1\}$ .

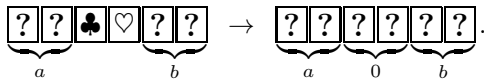
As will be seen soon, the idea behind the AND protocol described below is based on Eq. (2). Hereafter, for two bits  $x$  and  $y$ , the notation



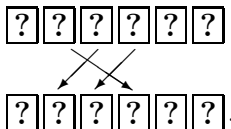
means



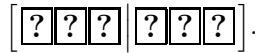
1. Arrange the six cards as follows:



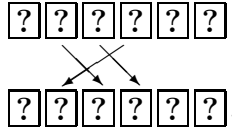
2. Rearrange the sequence of six cards as follows:



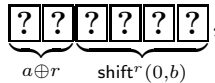
3. Apply a random bisection cut:



4. Rearrange the sequence of six cards as follows:

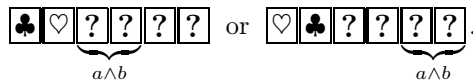


Then, we have



where  $r$  is a random bit because of the random bisection cut.

5. Reveal the leftmost two cards to find the value of  $a \oplus r$ , that tells us which commitment of the possible two is desired. (Recall Eq. (2), from which if  $a \oplus r = 0$ , then  $a \wedge b = \text{get}^0(\text{shift}^r(0, b))$  and otherwise  $a \wedge b = \text{get}^1(\text{shift}^r(0, b))$ .) Therefore, a commitment to  $a \wedge b$  is obtained as follows:



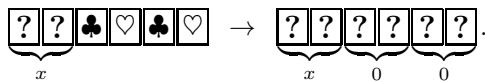
Note that revealing the leftmost commitment to  $a \oplus r$  in step 5 does not leak any information about  $a$  because  $r$  is random. In addition, the leftmost two face-up cards are available for another computation.

A six-card OR protocol can be easily constructed in a similar manner [6].

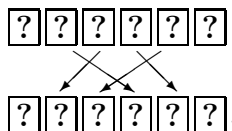
### 2.3 The Copy Protocol with a Random Bisection Cut

Given a commitment to a bit  $x$ , four additional cards are sufficient to make two copies of the commitment [8].

1. Arrange the four additional cards to the right of the given commitment:

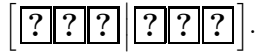


2. Rearrange the sequence of six cards as follows:

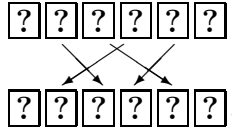




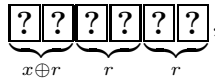
3. Apply a random bisection cut:



4. Rearrange the sequence of six cards as follows:

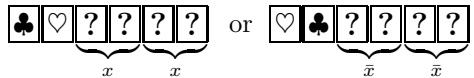


Then, we have



where  $r$  is a random bit because of the random bisection cut.

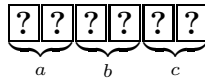
5. Reveal the leftmost two cards. Then, we have



Remember that a commitment to  $x$  is easily obtained from a commitment to  $\bar{x}$ .

### 3 Straightforward Secure Majority Computations

In this section, by applying the existing protocols (introduced in the previous section), we show that a three-input secure majority computation can be naively done with 14 cards. That is, given three commitments

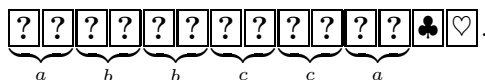


together with eight additional cards, we can obtain a commitment



to the majority value, as follows.

First, remember that applying the copy protocol mentioned in Section 2.3 to a commitment along with four additional cards results in two copied commitments as well as two available cards. Therefore, by applying the copy protocol three times, one at a time, we can copy the three commitments using eight additional cards so that we have



Next, since

$$\text{maj}(a, b, c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a),$$

we can easily obtain a commitment to  $\text{maj}(a, b, c)$  by applying the AND protocol and the OR protocol mentioned in Section 2.2. (Recall that the AND/OR protocol needs only two additional cards.)

Thus, a three-input secure majority computation can be straightforwardly conducted with eight additional cards. Alternatively, using the Full Adder protocol [6] (also shown in Table 1) with three input commitments and four additional cards, we can obtain a commitment to the carry  $(a \wedge b) \vee ((a \oplus b) \wedge c)$ , which is equal to  $\text{maj}(a, b, c)$ . In this paper, we improve the results further, that is, in the next section, we design a tailor-made protocol for a secure majority computation that is simple and needs only two additional cards.

## 4 An Improved Secure Majority Protocol

In this section, we propose an efficient and simple secure majority protocol that requires six fewer cards than the naive method presented above (and two fewer cards than the Full Adder protocol [6]).

In Section 4.1, we first introduce the idea behind our new protocol. We then describe our protocol in Section 4.2.

### 4.1 The Idea

Given three bits  $a, b, c \in \{0, 1\}$ , if  $a = b$ , then  $\text{maj}(a, b, c)$  is equal to  $a$ ; otherwise,  $\text{maj}(a, b, c)$  is determined by  $c$ . Therefore, it holds that

$$\text{maj}(a, b, c) = \text{get}^{a \oplus b}(a, c).$$

Hence, our protocol first makes the following sequence:

$$\underbrace{\boxed{?} \boxed{?}}_{a \oplus b} \underbrace{\boxed{?} \boxed{?}}_a \underbrace{\boxed{?} \boxed{?}}_c.$$

If we turned over the leftmost two cards, then we could determine the position of the desired commitment to  $\text{get}^{a \oplus b}(a, c)$ , but the value of  $a \oplus b$  would also be leaked. Therefore, our protocol next adds randomization to hide the value of  $a \oplus b$ , in a manner similar to how the six-card AND protocol shown in Section 2.2 does: it produces a sequence

$$\underbrace{\boxed{?} \boxed{?}}_{a \oplus b \oplus r} \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\text{shift}^r(a, c)}$$

where  $r$  is a random bit. Based on the equality

$$\text{maj}(a, b, c) = \text{get}^{a \oplus b \oplus r}(\text{shift}^r(a, c)),$$

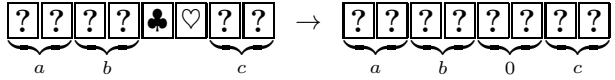
we can obtain a commitment to  $\text{maj}(a, b, c)$  by revealing the leftmost two cards.

The full description of our protocol is presented in the next subsection.

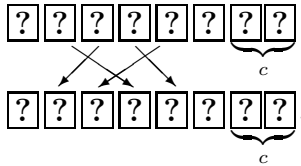
### 4.2 An Eight-Card Secure Majority Protocol

Given three commitments to bits  $a$ ,  $b$  and  $c$  together with two additional cards, our secure majority protocol proceeds as follows.

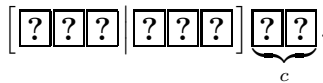
1. Arrange the three commitments and the two additional cards as shown below:



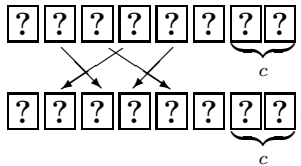
2. Rearrange their order:



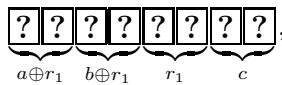
3. Apply a random bisection cut to the leftmost six cards:



4. Rearrange the order again:

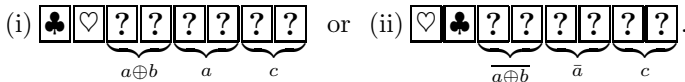


Then, we have

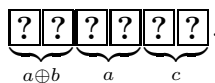


where  $r_1$  is a random bit because of the random bisection cut.

5. Reveal the leftmost two cards. Then, we have either

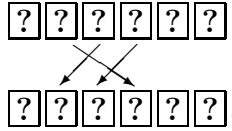


In case (ii), apply the secure NOT operation to each of the commitments to  $\overline{a \oplus b}$  and  $\bar{a}$ . Hence, in either case, we have three commitments

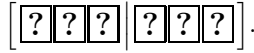


From now on, we are working only on the six face-down cards above.

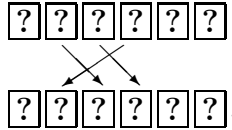
6. Rearrange the sequence of six cards as follows:



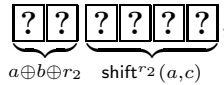
7. Apply a random bisection cut:



8. Rearrange the order again:

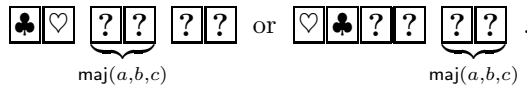


Then, we have



where  $r_2$  is a random bit because of the random bisection cut.

9. Reveal the leftmost two cards to find the value of  $a \oplus b \oplus r_2$ . Then, a commitment to  $\text{maj}(a, b, c)$  is obtained as follows:



## 5 Conclusion

In this paper, we designed an eight-card three-input secure majority protocol whose output is in a committed format. Since the naive implementation of a secure majority computation requires 14 cards, we have succeeded in reducing the number of required cards by six. Furthermore, our protocol is simple and easy to understand.

Generally, when a function  $f$  can be securely computed with some number of cards, any function derived from  $f$  by (i) negation of variables, (ii) permutation of variables or (iii) negation of  $f$  can also be securely computed with the same number of cards. Therefore, when focusing on the three-variable symmetric Boolean functions, it is sufficient to consider the following six representatives of the “NPN-equivalence [12]” classes:

$$S_{\emptyset}^3, S_{\{3\}}^3, S_{\{1,2\}}^3, S_{\{1,3\}}^3, S_{\{2,3\}}^3, S_{\{0,2,3\}}^3$$

where  $S_X^3$  for  $X \subseteq \{0, 1, 2, 3\}$  represents a three-variable symmetric Boolean function such that

$$S_X^3(a, b, c) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } a + b + c \in X; \\ 0 & \text{otherwise.} \end{cases}$$

Note that  $S_{\{2,3\}}^3(a, b, c) = \text{maj}(a, b, c)$ . There is nothing to do with  $S_{\emptyset}^3 = 0$ . Since  $S_{\{3\}}^3(a, b, c) = a \wedge b \wedge c$  and  $S_{\{1,3\}}^3(a, b, c) = a \oplus b \oplus c$ , they can be securely computed with eight cards and six cards, respectively. The remaining two functions  $S_{\{1,2\}}^3$  and  $S_{\{0,2,3\}}^3$  can also be securely computed with eight cards by using the equations:

$$S_{\{1,2\}}^3(a, b, c) = \text{get}^{a \oplus b}(a \oplus c, 1)$$

and

$$S_{\{0,2,3\}}^3(a, b, c) = \text{get}^{a \oplus b \oplus c}(1, a \wedge c).$$

Thus, any three-variable symmetric function can be securely computed with eight cards or less: that is, within two additional cards. An intriguing open question is whether any four-variable (or larger) symmetric Boolean function can be securely computed within two additional cards. Furthermore, it is also interesting to examine the question of whether there exists a “non-committed format” majority protocol that requires no additional card.

Aside from the practical importance of implementing secure computations, the research area of card-based protocols along with other physically implemented cryptographic protocols (e.g. [1,4,10]) can aid professional cryptographers in providing intuitive explanations to non-specialists about what cryptography is in general. We believe that such protocols will also be helpful for information science education in the classroom.

**Acknowledgments.** This work was supported by JSPS KAKENHI Grant Numbers 23700007 and 25289068.

## References

- Balogh, J., Csirik, J.A., Ishai, Y., Kushilevitz, E.: Private computation using a PEZ dispenser. *Theoretical Computer Science* 306(1–3), 69–84 (2003)
- den Boer, B.: More efficient match-making and satisfiability: the five card trick. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 208–217. Springer, Heidelberg (1990)
- Crépeau, C., Kilian, J.: Discreet solitary games. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 319–330. Springer, Heidelberg (1994)
- Fagin, R., Naor, M., Winkler, P.: Comparing information without leaking it. *Communications of the ACM* 39(5), 77–85 (1996)
- Goldreich, O.: *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York (2004)
- Mizuki, T., Asiedu, I.K., Sone, H.: Voting with a logarithmic number of cards. In: Mauri, G., Dennunzio, A., Manzoni, L., Porreca, A.E. (eds.) UCNC 2013. LNCS, vol. 7956, pp. 162–173. Springer, Heidelberg (2013)
- Mizuki, T., Kumamoto, M., Sone, H.: The five-card trick can be done with four cards. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 598–606. Springer, Heidelberg (2012)
- Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) FAW 2009. LNCS, vol. 5598, pp. 358–369. Springer, Heidelberg (2009)

9. Mizuki, T., Uchiike, F., Sone, H.: Securely computing XOR with 10 cards. *The Australasian Journal of Combinatorics* 36, 279–293 (2006)
10. Moran, T., Naor, M.: Polling with physical envelopes: A rigorous analysis of a human-centric protocol. In: Vaudenay, S. (ed.) *EUROCRYPT 2006*. LNCS, vol. 4004, pp. 88–108. Springer, Heidelberg (2006)
11. Niemi, V., Renvall, A.: Secure multiparty computations without computers. *Theoretical Computer Science* 191(1–2), 173–183 (1998)
12. Sasao, T.: *Switching Theory for Logic Synthesis*, 1st edn. Kluwer Academic Publishers, Norwell (1999)
13. Schneider, T.: *Engineering Secure Two-Party Computation Protocols: Design, Optimization, and Applications of Efficient Secure Function Evaluation*. Springer, Heidelberg (2012)
14. Stiglic, A.: Computations with a deck of cards. *Theoretical Computer Science* 259(1–2), 671–678 (2001)
15. Yao, A.C.: Protocols for secure computations. In: *Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science, FOCS 1982*, pp. 160–164. IEEE Computer Society, Washington, DC (1982)

# An Analysis of a Selecto-Lamarckian Model of Multimemetic Algorithms with Dynamic Self-organized Topology

Rafael Nogueras<sup>1</sup>, Carlos Cotta<sup>1</sup>, Carlos M. Fernandes<sup>2</sup>,  
Juan Luis Jiménez Laredo<sup>3</sup>, Juan Julián Merelo<sup>4</sup>, and Agostinho C. Rosa<sup>2</sup>

<sup>1</sup> Dept. Lenguajes y Ciencias de la Computación, Universidad de Málaga, Spain

<sup>2</sup> Systems and Robotics Institute, Technical University of Lisbon, Portugal

<sup>3</sup> University of Luxembourg, Luxembourg

<sup>4</sup> Dept. Arquitectura y Tecnología de Computadores, Universidad de Granada, Spain  
ccottap@lcc.uma.es

**Abstract.** Multimemetic algorithms (MMAs) are memetic algorithms that explicitly represent and evolve memes (computational representations of problem solving methods) as a part of solutions. We use an idealized selecto-Lamarckian model of MMAs in order to analyze the propagation of memes in spatially structured populations. To this end, we focus on the use of dynamic self-organized spatial structures, based on the stimergetic communication among solutions, and compare these with regular static lattices and unstructured (panmictic) populations. An empirical analysis indicates that these dynamic lattices are capable of promoting memetic diversity and provide better results in terms of survival of high-quality memes.

**Keywords:** Memetic algorithms, spatial structure, self-organization.

## 1 Introduction

The paradigm of memetic optimization was conceived 25 years ago [14] as a pragmatic combination of ideas from population-based global search techniques and trajectory-based local search techniques [9,16,17]. Ideas from cultural evolution or Lamarckian lifetime learning [3] are usually cited in connection with memetic algorithms (MAs) and –while some times overstressed– they provide a useful metaphor for framing the functioning of these techniques or for building actual algorithmic incarnations of MAs. Central to this metaphor is the notion of *meme*. Defined four decades ago by Richard Dawkins [4] as an analogy to biological genes in the context of cultural evolution, memes can be thought of as units of imitation, i.e., ideas or pieces of knowledge that jump from individual to individual, proliferating or becoming extinct depending on the benefit they provide to their hosts and/or the compulsion the latter feel to propagate them. Quite interestingly, memes also exhibit a high adaptability, being able to change during the lifetime of the individual, thus evolving separately from the biological substrate they depend on.

The idea of exploiting the evolution of memes in a computational framework for optimization was already anticipated in some early works on the topic [15] and is now a core idea in the concept of memetic computing (MC) [19], which is defined as “...a paradigm that uses the notion of meme(s) as units of information encoded in computational representations for the purpose of problem solving”, see [17]. Such an explicit treatment of memes can be found in multimemetic algorithms (MMAs) [10,11], in which each solution carries memes indicating how to perform local optimization on it.

An important issue in MMAs is the way in which memes propagate through the population. In this sense, note that the dynamics of meme propagation are more complex than that of their genetic counterparts (the latter being a topic that has been thoroughly studied, e.g., [2,8,20,21]). The main reason is the fact that while genes represent solutions whose quality is objectively measurable via the fitness function, memes are only indirectly evaluated according to the effect they exert on the solutions they are attached to. Hence, a mismatch between genes and memes may cause potentially good memes be ignored by the selection operator, or the other way around, bad memes may proliferate if they are lucky to attach to good solutions. A first analysis in this direction was done in [18], in which the previous issues were theoretically established using a simplified selecto-Lamarckian model of MMAs, whose dynamics was later studied in the context of panmictic (unstructured) and spatially-structured populations. The higher diversity and longer takeover times induced by the latter were essential to allow good memes to express themselves in the population and survive up to the final stages of evolution. An overview of this model is provided in Sect. 3.

The importance of population structure in this context highlights the need for analyzing different models for structuring the population. Regular static lattices with von Neuman topology were considered in [18]. Here, we turn our attention to non-regular dynamic structures –a brief overview of some of such structures is provided in Sect. 2. To be precise, we consider the self-organized model proposed by Fernandes et al. [5]. This model will be deployed on the selecto-Lamarckian model of MMAs, and the effect it produces on the propagation of memes will be empirically analyzed in Sect. 4. We will close with conclusions and an outline of future research directions in Sect. 5.

## 2 Background

Dynamic population structures have been attracting the interest of the research community in recent times. For example, Alba and Dorronsoro [1] proposed a mechanism for adapting dynamically the radius of the neighborhood under a fixed underlying topology. Then, Whiteacre et al. [22] proposed a dynamic structure featuring self-organized definition of locality and interaction epistasis in order to sustain genetic diversity. Also, Laredo et al. [12,13] analyzed EAs working on peer-to-peer environments whose topology is inherently dynamic and volatile, showing how these EAs are more scalable and resilient to failures.

A particularly interesting model was defined by Fernandes et al. [5] combining ideas from swarm intelligence and cellular automata. This model –which can



be regarded as a cellular automata with short-term memory— uses stimergetic communication and simple rules for movement on a large 2D-lattice, giving rise to self-organized clusters of particles. A noticeable feature of these clusters is that they keep evolving and changing shape, thus providing some kind of highly dynamic order.

This latter model can be naturally deployed on metaheuristics –indeed, highly promising results have been obtained from its application to particle-swarm-optimization algorithms [6,7]– since the spatial location of particles (individuals or solutions) can be used as the basis for defining the population neighborhood structure to which operations such as selection or recombination are constrained. The neighborhood relationship is therefore dynamic and self-organized by virtue of the movement rules, which aim to place particles close to other similar particles (similarity is here defined in terms of fitness, but any alternative definition could be used in principle). The resulting dynamics resemble in some sense a mixture of island-based EAs and cellular EAs, since clusters can be regarded as complex islands among which information flows via migrating particles or paths thereof. It is thus of the foremost interest to analyze how memes would propagate in such an environment, and study whether it constitutes a viable alternative to static population structures. Subsequent sections are devoted to this.

### 3 Model Description

As stated in Sect. 1, we consider an idealized model of MMAs which we shall augment with a self-organized dynamic topology. Thus, let us firstly describe the underlying model in Sect. 3.1. Subsequently, the extension to dynamic topology will be described in Sect. 3.2.

#### 3.1 The Selecto-Lamarckian Model

Let us consider an abstract characterization of MMAs consisting of a population  $P = [\langle g_1, m_1 \rangle, \dots, \langle g_\mu, m_\mu \rangle]$  of  $\mu$  individuals, which are subject to the application of evolutionary operators for selection, local search and replacement as shown in Algorithm 1. Each of the individuals in the population is a tuple  $\langle g_i, m_i \rangle \in D^2$ , for some  $D \subset \mathbb{R}$ . The first component of the tuple – $g$ – represents the genotype (which for simplicity we shall equate to fitness). As to the second component – $m$ – it represents a meme or, more precisely, the potential of the meme, that is, how good solutions can become by using this meme. This potential is exercised via a monotonic increasing function  $f : D^2 \rightarrow D$  which captures the application of a meme to a gene: an individual  $\langle g, m \rangle$  becomes  $\langle f(g, m), m \rangle$  after the application of the meme, where

$$\lim_{n \rightarrow \infty} f^n(g, m) = m \text{ if } g < m \tag{1}$$

$$f(g, m) = g \text{ if } g \geq m \tag{2}$$

Here  $f^n(g, m)$  is the  $n$ -fold composition of  $f$  on its first argument. Thus, the meme would leave unchanged a solution which is better than the former, and

**Algorithm 1.** Selecto-Lamarckian Model

---

```

for  $i \in [1 \cdots \mu]$  do
  | INITIALIZE( $g_i, m_i$ );
end
while  $\neg$  CONVERGED ( $P$ ) do
  |  $i \leftarrow$  URAND( $1, \mu$ ) // Pick random location
  |  $\langle g, m \rangle \leftarrow$  SELECTION( $P, S, i$ );
  |  $g' \leftarrow f(g, m)$  // Local improvement
  |  $P \leftarrow$  REPLACE( $P, S, i, \langle g', m \rangle$ );
end

```

---

would otherwise improve the latter, reaching its potential in the limit. While this is a highly idealized description of the action of memes (which in general will depend on the match between the genotype and the meme on a problem-specific basis) it constitutes an initial approximation that can be used to study the generalities of meme propagation.

Interaction among individuals is restricted by a spatial structure given by a  $\mu \times \mu$  Boolean matrix  $S$ , where  $S_{ij} = \mathbf{true}$  if, and only if, the individual in the  $i$ -th location can interact with the individual in the  $j$ -th location. Two static possibilities were considered in [18], namely the panmictic case (i.e.,  $S_{ij} = \mathbf{true}$  for all  $i, j$ ) and a square lattice with von Neumann topology (i.e.,  $S_{ij} = \mathbf{true}$  iff  $L_1(i, j) \leq r$ , where  $L_1(\cdot, \cdot)$  is the Manhattan distance and  $r$  is the neighborhood radius). Next section considers the use of a dynamic self-organized population structure.

### 3.2 Dynamic Self-organized Topology

Let us consider a rectangular grid  $G$  of size  $r \times s > \mu$ . Each cell  $G_{uv}$  of the grid is a tuple  $(\eta_{uv}, \zeta_{uv})$ , where  $\eta_{uv} \in \{1, \dots, \mu\} \cup \{\bullet\}$  and  $\zeta_{uv} \in (D \times \mathbb{N}) \cup \{\bullet\}$ . The value  $\eta_{uv}$  indicates the index of the individual that occupies position  $\langle u, v \rangle$  in the grid. If  $\eta_{uv} = \bullet$  the corresponding position is empty but could still have some information, namely a mark  $\zeta_{uv}$  (if  $\zeta_{uv} = \bullet$  the position is empty and devoid of any mark). Such marks are placed by individuals that were occupying that position in the past and consist of the fitness value  $\zeta_{uv}^f$  of the individual and a time stamp  $\zeta_{uv}^t$  that indicates the iteration in which the mark was placed. All marks have a lifespan of  $K$  iterations after being placed, and are deleted afterwards.

Initially  $G_{uv} = (\bullet, \bullet)$  for all  $\langle u, v \rangle$ . Then, individuals are placed in random positions of the grid (only one individual per cell is allowed at most). These initial positions define the initial value of the Boolean matrix  $S$  as follows: let  $\rho : \mathbb{N} \rightarrow \mathbb{N}^2$  be a function that returns the position  $\langle u, v \rangle$  a certain individual is in; then  $S_{ij} = \mathbf{true}$  iff  $L_\infty(\rho(i), \rho(j)) \leq r$ , where  $L_\infty(\cdot, \cdot)$  is the maximum distance and  $r$  is the neighborhood radius, i.e., the Moore neighborhood of radius  $r$ , following [5].

---

**Algorithm 2.** Individual Movement ( $i, t, G$ )

---

```

 $\langle u, v \rangle \leftarrow \rho(i)$ ;  $move \leftarrow true$ ;
if exists  $\langle u^{(i)}, v^{(i)} \rangle \in \mathcal{N}\langle u, v \rangle$  such that  $\zeta_{uv}^f > g_i$  then
    |  $\langle u', v' \rangle \leftarrow \arg \min\{\zeta_{uv}^f \mid \zeta_{uv}^f > g_i\}$ ;
else
    | if exists  $\langle u^{(i)}, v^{(i)} \rangle \in \mathcal{N}\langle u, v \rangle$  such that  $\zeta_{uv}^f < g_i$  then
        | |  $\langle u', v' \rangle \leftarrow \arg \max\{\zeta_{uv}^f \mid \zeta_{uv}^f < g_i\}$ ;
        | else
            | if  $\mathcal{N}\langle u, v \rangle \neq \emptyset$  then
                | | Pick  $\langle u', v' \rangle$  at random from  $\mathcal{N}\langle u, v \rangle$ ;
            | else
                | |  $move \leftarrow false$ ;
            | end
        | end
    | end
end
if  $move$  then
    |  $\zeta_{uv}^f \leftarrow g_i$ ;  $\zeta_{uv}^t \leftarrow t$ ; // mark old cell
    |  $\eta_{uv} = \bullet$ ;  $\eta_{u'v'} = i$ ; // move to new cell
end

```

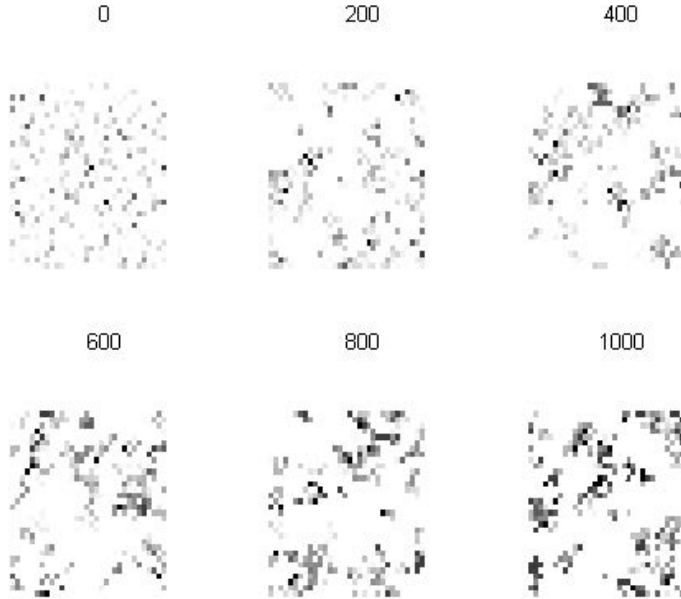
---

Subsequently –in each iteration, after all individuals have been subject to the evolutionary operators– each individual moves to an adjacent empty position. Adjacency is again defined on the basis of the Moore neighborhood of radius  $r$ , so an individual  $i$  placed at  $\rho(i) = \langle u, v \rangle$  can move to an empty position  $\langle u', v' \rangle$  for which  $L_\infty(\langle u, v \rangle, \langle u', v' \rangle) \leq r$ . If no empty position is available, the individual rests at its position. Otherwise, it picks a neighboring empty cell according to the marks on them. More precisely, let  $\mathcal{N}\langle u, v \rangle = \{\langle u^{(1)}, v^{(1)} \rangle, \dots, \langle u^{(w)}, v^{(w)} \rangle\}$  be the collection of empty neighboring cells and let  $i$  be the index of the individual to move. Then, the new cell is selected as indicated in Algorithm 2, i.e., it attempts to move to a cell whose mark is as close as possible to its own fitness (first from above, then from below) or to a random cell if no such mark exists.

Once individuals have moved, matrix  $S$  is updated. Fig. 1 shows an example of how individuals distribute on the grid as iterations go by. Notice how starting from a random distribution they start to quickly group in clusters and how these evolve in shape during the run. Next section will empirically analyze how the use of this dynamic topology influences the propagation of memes.

## 4 Experimental Analysis

In order to analyze the behavior of our model, we have considered the following parameterization: selection is done using binary tournament with probability  $p_S$  (if the random test is not passed the offspring is a copy of the initial parent); local search is implemented using the following function

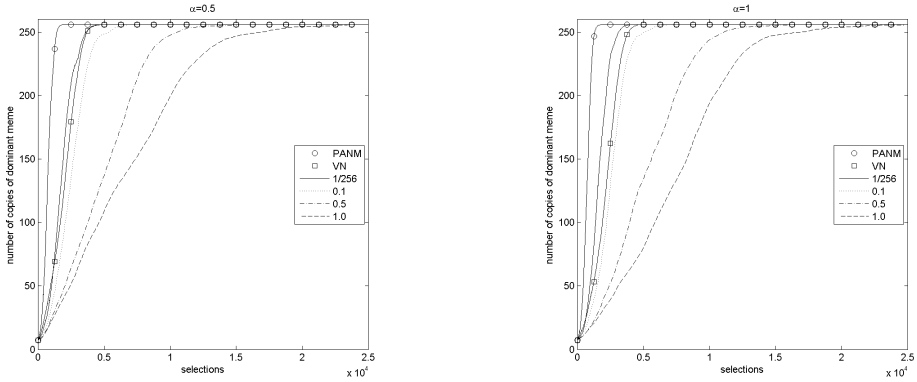


**Fig. 1.** Distribution of individuals on the grid ( $\mu = 256$ ,  $r = s = 32$ ) after a number of iterations. Gray shades indicate the meme potential (darker is better).

$$f(g, m) = \begin{cases} g & \text{if } g \geq m \\ (g + m)/2 & \text{if } g < m \end{cases} \quad (3)$$

which intuitively provides increasingly smaller improvements on increasingly better solutions as it typically happens in practice. The function is applied with probability  $p_{LS}$  (if the random test is not passed the offspring is left unchanged); replacement is done by substituting the parent who lost the tournament by the offspring. Numerically, we consider  $\mu = 256$ ,  $r = 1$ ,  $K = 1$  and  $p_S = p_{LS} \in \{1/256, 0.1, 0.5, 1.0\}$ . All individuals are initialized at random: memes are drawn from  $[0, 1]$  and genes from  $[0, \alpha]$ , where we consider  $\alpha \in \{0.5, 1.0\}$ . The purpose of this parameter  $\alpha$  is to tune the relative capacity of improvement that memes have. Intuitively, larger values of  $\alpha$  make it more likely that a low-potential meme can attach to a high-quality solution, thus implying a harder survival scenario for high-potential memes. We conduct series of twenty five runs per configuration. Each run ends when the population has converged and all memes are equal to two decimal positions.

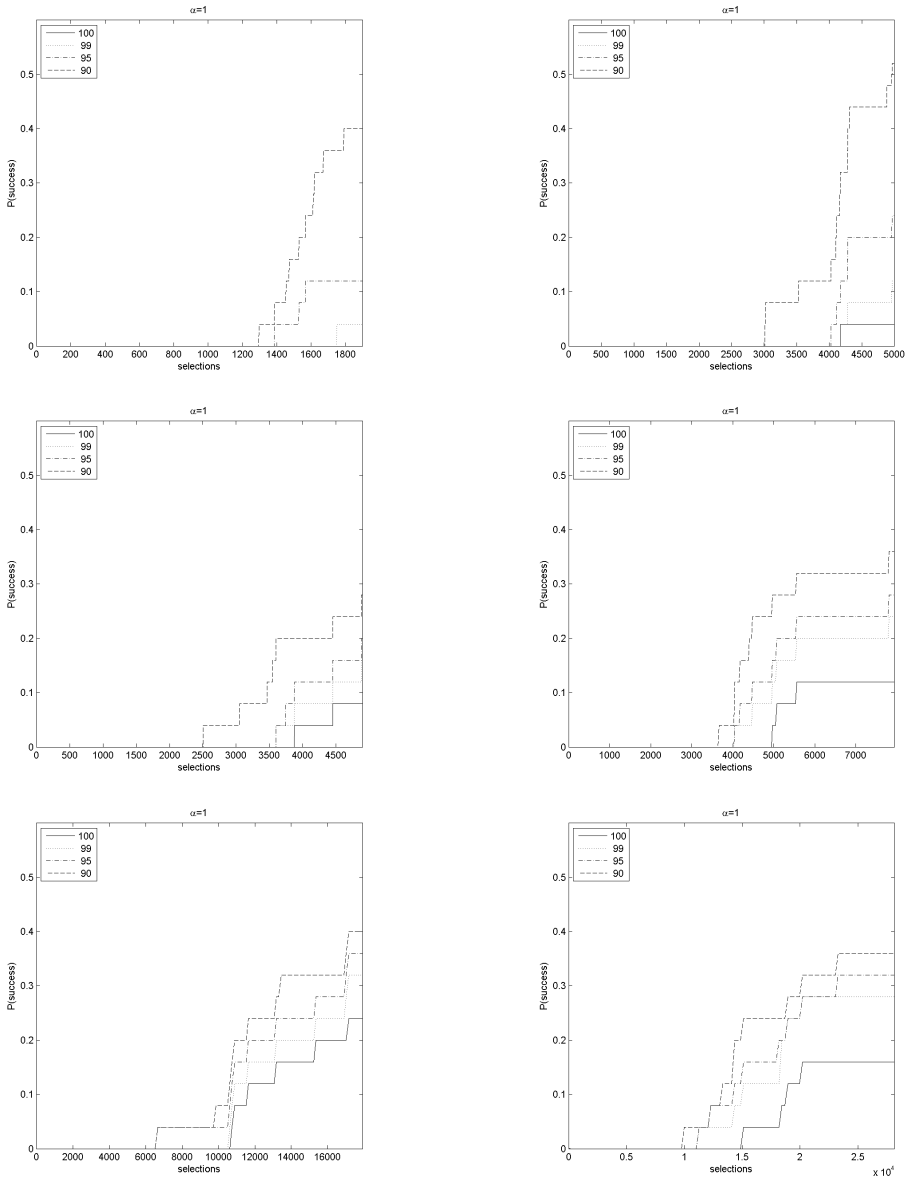
Let us firstly analyze how diversity is maintained in each of the configurations. Fig. 2 shows how the number of copies of the dominant meme (i.e., the meme with the higher number of copies, not necessarily the same one in each moment) grows until taking over the population. As expected, the panmictic model is the first to converge whereas models with spatial structure manage to keep diversity



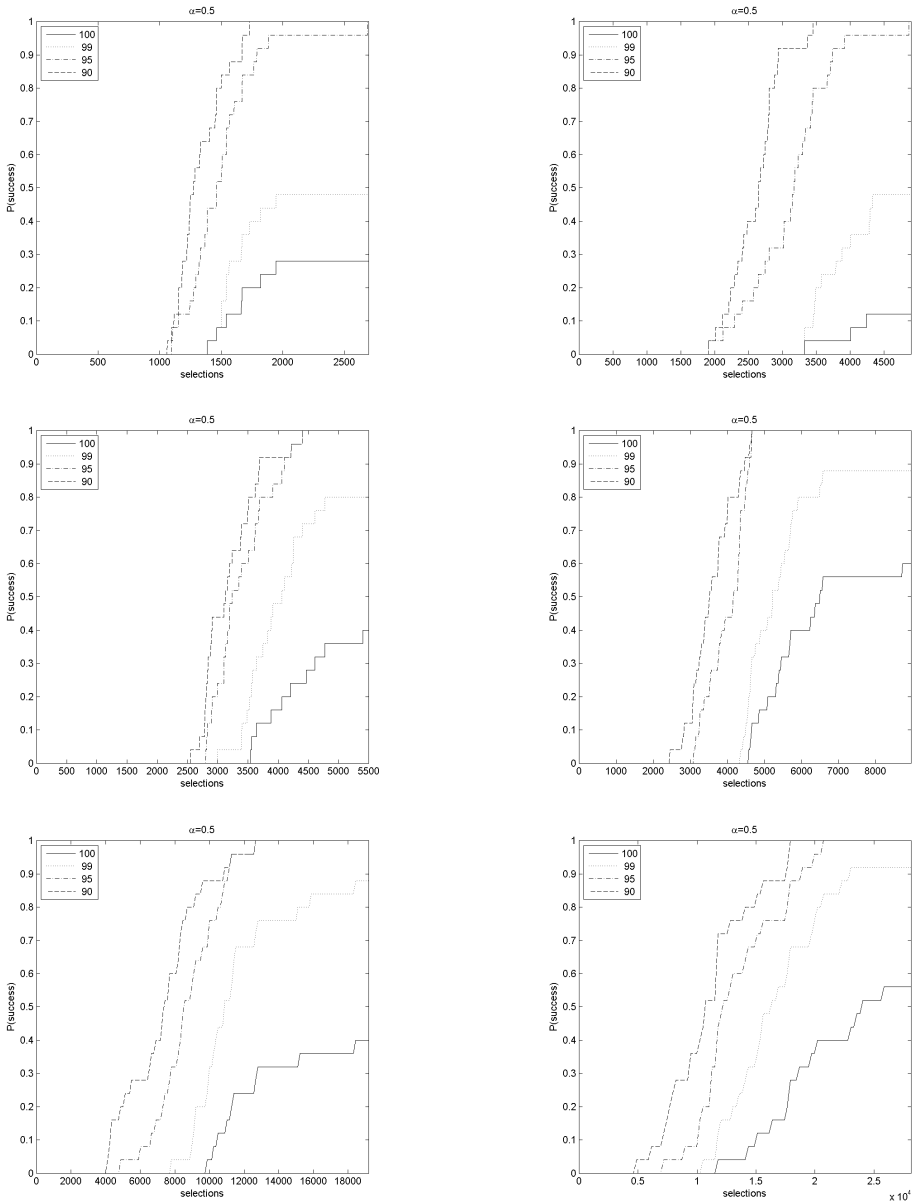
**Fig. 2.** Growth curves for different configurations. (Left)  $\alpha = 0.5$ . (Right)  $\alpha = 1$ .

for a longer time; the dynamic model with low values of  $p_S$  behaves similarly to the static von Neumann model, but high values of  $p_S$  result in slower convergence. This may seem counter-intuitive but admits an explanation. Notice that reducing the value of  $p_S$  can be also regarded as increasing the ratio of self-organization to evolution (individuals are given more time to move and self-organize between the actual application of the selection operator). This means on one hand that they can group together and hence selection is more effective than if done on isolated individuals on the grid. On the other hand, individuals flow across the grid and hence can communicate genetic information at a faster pace than on a static topology.

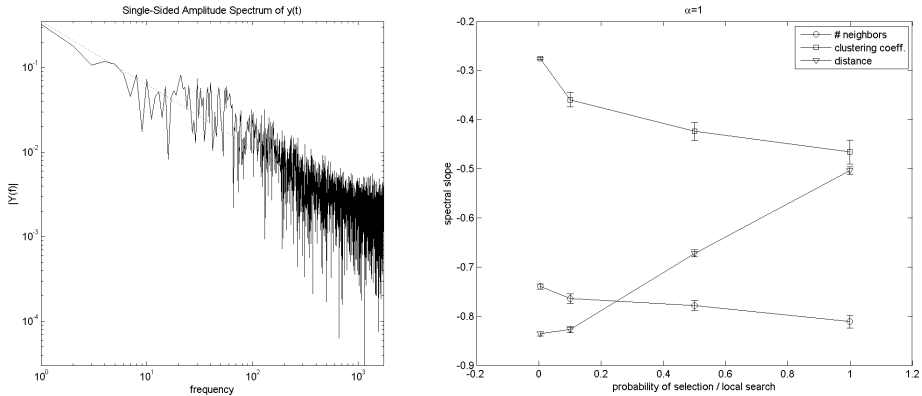
Let us now turn our attention to the effect this slower convergence has on the survival rates of high-potential memes. Figs. 3–4 provide qualified run time distributions, indicating the percentage of runs that reach a certain goal (in this case, the population being dominated by a meme within a certain percentile of the initial meme distribution) as a function of the number of selection operations performed. Not surprisingly –and as shown in [18]– instances with  $\alpha = 0.5$  converge to high-quality memes much more often than for  $\alpha = 1$ , the reason being that it is much more difficult for low-potential memes to hitchhike their way to the final population by attaching to good solutions. Also, models with spatial structure behave better than the panmictic model in this case due to the fact that the slower convergence rate buys time for high-potential memes to express themselves in the population by improving their hosts and hence increase their survival rates. This is also reflected in the models with dynamic topology, in particular in those with larger values of  $p_S$ , which can clearly improve the success rate of the static von Neumann model. Very high-quality memes can make it to the final stages of evolution in the dynamic model in contrast with the comparatively higher extinction rates such memes exhibit in panmictic populations and in the static von Neumann lattice.



**Fig. 3.** Qualified runtime distributions for  $\alpha = 1$ . From top to bottom and left to right: panmictic population, von Neuman topology,  $p_S = 1/256$ ,  $p_S = 0.1$ ,  $p_S = 0.5$  and  $p_S = 1.0$ . The X scale corresponds to the number of evaluations to convergence in each case.



**Fig. 4.** Qualified runtime distributions for  $\alpha = 0.5$ . From top to bottom and left to right: panmictic population, von Neuman topology,  $p_s = 1/256$ ,  $p_s = 0.1$ ,  $p_s = 0.5$  and  $p_s = 1.0$ .



**Fig. 5.** (Left) Spectrum of the average number of neighbors ( $p_S = 1/256$ ). (Right) Mean slope of the spectrum. The error bars indicate the standard error of the mean. In both cases,  $\alpha = 1$ .

We have finally conducted a spectral analysis of the evolution of different properties of the population structure along time. These properties are: (i) the average number of neighbors each individual has, (ii) the average clustering coefficient of the neighborhood graph (i.e., the average percentage of an individual's neighbors that are neighbors themselves), and (iii) the average genotypic distance among neighbors. For each of these time-dependent magnitudes we have computed the Fast Fourier Transform, and removed conjugated coefficients to obtain the frequency spectrum. An example of such a spectrum can be seen in Fig. 5 (left). As it can be seen, the intensity is proportional to  $f^a$  for some  $a < 0$ . The value of this parameter indicate a characteristic kind of noise. A value close to 0 indicates white noise, whereas close to  $-2$  indicates Brownian noise. For values in-between (in particular close to  $-1$ ) we obtain pink noise, one of the signatures of self-organized systems.

Fig. 5 (right) shows the values of  $a$  obtained for  $\alpha = 1$  (results for  $\alpha = 0.5$  are qualitatively similar and not shown due to space limitations). In general, the spectrum slope is closer to pink noise in the number of neighbors. We hypothesize the different behavior of the distance is due to the fact that increasingly larger application rates of local search result in a very fast reduction of distances, which tend to be more homogeneous, and hence the fluctuations look closer to white noise. As to the clustering coefficient, it follows the same trend as the number of neighbors although with smaller magnitudes. This might be due to the fact that the clustering coefficient is a higher-level property than mere neighborhood (i.e., it is not a property among any two individuals but among two individuals that are simultaneously neighbors of a third one) and hence the underlying self-organization is only indirectly shown at this level.



## 5 Conclusions

We have analyzed the propagation of memes in a selecto-Lamarckian model of MMAs endowed with dynamic spatial structure. We have shown that a self-organized model based on stimergetic communication provides very promising results in comparison to unstructured populations and to populations arranged in static lattices. By tuning the ratio between self-organization (i.e., moves of individuals on the underlying grid) and evolutionary operators (i.e., selection and local search) the convergence of the algorithm can be adjusted to be slower/faster. For some low rates of application of evolutionary operators, the resulting model is similar in convergence speed to a static von Neumann lattice, yet still manages to slightly improve the success rate. The difference is more marked for higher rates of application of selection/local search, resulting in much better success rates as indicated by the qualified runtime distributions.

Lines for future research are manifold. Firstly, testing other topologies and movement policies is important in order to elucidate whether the observed behavior is due to the particular stimergetic model considered or just emanates from the dynamism of the topology. In addition, it is worth studying what the behavior of the system is if the neighborhood used for movement is decoupled from the neighborhood used for evolutionary interaction, actually utilizing different neighborhoods for either purpose. Last but not least, it is clear that the models described in this work should be eventually put at work and analyzed in a full-fledged optimization environment.

**Acknowledgements.** This work is partially supported by MICINN project ANY-SELF (TIN2011-28627-C04-01/-02), by Junta de Andalucía projects P08-TIC-03903 and P10-TIC-6083 (DNEMESIS), by project CANUBE (CEI2013-P-14) and by Universidad de Málaga, Campus de Excelencia Internacional Andalucía Tech.

## References

1. Alba, E., Dorronsoro, B.: The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation* 9, 126–142 (2005)
2. Alba, E., Luque, G.: Growth curves and takeover time in distributed evolutionary algorithms. In: Deb, K., Tari, Z. (eds.) *GECCO 2004*. LNCS, vol. 3102, pp. 864–876. Springer, Heidelberg (2004)
3. Castillo, P., Arenas, M.G., Castellano, J.G., Merelo, J.J., Prieto, A., Rivas, R.: Lamarckian evolution and the Baldwin effect in evolutionary neural networks. In: Alba, E., et al. (eds.) *AEB 2002*, pp. 284–293. Universidad de Extremadura (2002)
4. Dawkins, R.: *The Selfish Gene*. Clarendon Press, Oxford (1976)
5. Fernandes, C.M., Merelo, J., Cotta, C., Rosa, A.: Towards a 2-dimensional self-organized framework for structured population-based metaheuristics. In: *2012 International Conference on Complex Systems*, pp. 1–6. IEEE Press (2012)
6. Fernandes, C.M., Guervós, J.J.M., Laredo, J.L.J., Cotta, C., Rosa, A.C.: Partially connected topologies for particle swarm. In: Blum, C., Alba, E. (eds.) *GECCO 2013 (Companion)*, pp. 11–12. ACM (2013)

7. Fernandes, C.M., Rosa, A.C., Laredo, J.L., Cotta, C., Merelo, J.J.: A study on time-varying partially connected topologies for the particle swarm. In: 2013 IEEE Congress on Evolutionary Computation (CEC), pp. 2450–2456 (2013)
8. Giacobini, M., Tomassini, M., Tettamanzi, A., Alba, E.: Selection intensity in cellular evolutionary algorithms for regular lattices. *IEEE Transactions on Evolutionary Computation* 9(5), 489–505 (2005)
9. Hart, W.E., Krasnogor, N., Smith, J.E.: Memetic Evolutionary Algorithms. In: Hart, W.E., Smith, J., Krasnogor, N. (eds.) *Recent Advances in Memetic Algorithms*. STUDEFUZZ, vol. 166, pp. 3–27. Springer-, Heidelberg (2005)
10. Krasnogor, N., Blackburne, B., Burke, E., Hirst, J.: Multimeme algorithms for protein structure prediction. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacanas, J.-L., Schwefel, H.-P. (eds.) *PPSN 2002*. LNCS, vol. 2439, pp. 769–778. Springer, Heidelberg (2002)
11. Krasnogor, N., Gustafson, S.: A study on the use of “self-generation” in memetic algorithms. *Natural Computing* 3(1), 53–76 (2004)
12. Laredo, J.L.J., Castillo, P.A., Mora, A.M., Merelo, J.J., Fernandes, C.M.: Resilience to churn of a peer-to-peer evolutionary algorithm. *International Journal of High Performance Systems Architecture* 1(4), 260–268 (2008)
13. Laredo, J.L.J., Eiben, A.E., van Steen, M., Merelo, J.J.: Evag: A scalable peer-to-peer evolutionary algorithm. *Genetic Programming and Evolvable Machines* 11(2), 227–246 (2010)
14. Moscato, P.: On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Tech. Rep. Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA (1989)
15. Moscato, P.: Memetic algorithms: A short introduction. In: Corne, D., Dorigo, M., Glover, F. (eds.) *New Ideas in Optimization*. McGraw-Hill’s Advanced Topics In Computer Science Series, pp. 219–234. McGraw-Hill, London (1999)
16. Moscato, P., Cotta, C.: A gentle introduction to memetic algorithms. In: Glover, F., Kochenberger, G. (eds.) *Handbook of Metaheuristics*. International Series in Operations Research & Management Science, vol. 57, pp. 105–144. Kluwer Academic Press, New York (2003)
17. Neri, F., Cotta, C., Moscato, P.: *Handbook of Memetic Algorithms*. SCI, vol. 379. Springer, Heidelberg (2012)
18. Nogueras, R., Cotta, C.: Analyzing meme propagation in multimemetic algorithms: Initial investigations. In: *Federated Conference on Computer Science and Information Systems*, Cracow, Poland (forthcoming, 2013)
19. Ong, Y.S., Lim, M.H., Chen, X.: Memetic computation—past, present and future. *IEEE Computational Intelligence Magazine* 5(2), 24–31 (2010)
20. Rudolph, G., Sprave, J.: A cellular genetic algorithm with self-adjusting acceptance threshold. In: *1st IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, London, UK, pp. 365–372 (1995)
21. Sarma, J., De Jong, K.: An analysis of local selection algorithms in a spatially structured evolutionary algorithm. In: Bäck, T. (ed.) *7th International Conference on Genetic Algorithms*, pp. 181–186. Morgan Kaufmann (1997)
22. Whitacre, J.M., Sarker, R.A., Pham, Q.: The self-organization of interaction networks for nature-inspired optimization. *IEEE Transactions on Evolutionary Computation* 12, 220–230 (2008)

# Parallel Multi-Objective Genetic Algorithm GPU Accelerated Asynchronously Distributed NSGA II

Oliver Rice, Robert E. Smith, and Rickard Nyman

University College London (UCL)  
London WC1E 6BT, UK  
Oliver@oliverrice.com,  
Robert.Elliott.Smith@gmail.com,  
RickardNyman@gmail.com

**Abstract.** Multi-objective optimization problems consist of numerous, often conflicting, criteria for which any solution existing on the Pareto front of criterion trade-offs is considered optimal. In this paper we present a general-purpose algorithm designed for solving multi-objective problems (MOPS) on graphics processing units (GPUs). Specifically, a purely asynchronous multi-populous genetic algorithm is introduced. While this algorithm is designed to maximally utilize consumer grade nVidia GPUs, it is feasible to implement on any parallel hardware. The GPU's massively parallel architecture and low latency memory result in +125 times speed-up for proposed parametrization relative to single threaded CPU implementations. The algorithm, NSGA-AD, consistently solves for solution sets of better or equivalent quality to state-of-the-art methods.

**Keywords:** genetic algorithm, multi-objective optimization, GPU acceleration, parallel computing.

## 1 Introduction

Graphics Processing Units (GPUs), initially designed for graphics rasterisation, are increasingly receiving attention in the scientific computing community. This interest spurs from the massive raw compute capabilities associated with GPU's highly parallel architecture. The subsequent advent of general purpose GPU computing languages such as Compute Unified Device Architecture (CUDA) [13] and Open Compute Language (OpenCL) [12] have made GPU acceleration of computationally expensive algorithms an attractive prospect. Both CUDA and OpenCL are extensions of the C programming language.

The multiple layers of parallelism within genetic algorithms have made them prime candidates for GPU acceleration in the past. Beyond simply parallelizing the fitness function for each candidate solution, further efforts also parallelize selection, recombination and mutation operators [4]. Several categories of the resulting algorithms include island based [10], distributed [3], and cellular [2]. These algorithms have been designed and tested for various parallel hardware environments including CPU clusters, FPGAs [11] and GPU clusters [9] with

applications ranging from shop scheduling [1], biology [15], chemistry [18] and finance [17].

Despite proposals for parallelizing single-run multi-objective genetic algorithms [7], most multi-objective implementations either involve highly domain specific approaches [16,6] or restricted selection models. In these instances MOPS are decomposed to multiple scalar optimization problems (SOPs) and optimized independently [8]. The goal of this paper is to parallelize computation of the entire Pareto optimal frontier. To this end we propose a cooperative approach in which multiple populations are independently processed to find predefined subsections of the Pareto front. This is achieved through use of an innovative decomposition of fitness functions allowing implementation of an NSGA II variant for multiple asynchronously computed sub-populations. Each of these sub-populations is a speciating island which maps to a portion of the Pareto frontier. We call the algorithm nondominated sorting genetic algorithm asynchronously distributed (NSGA-AD).

In section 2 the GPU memory model is discussed. Following, in section 3 an outline of NSGA II is presented. Section 4 introduces the trade-offs associated with different GA memory mappings to the GPU. Section 5 combines the previous information in development of our proposed algorithm which is tested according to quality and speed in sections 6 and 7 respectively. We end with conclusions in section 8.

## 2 GPU Memory Model

General purpose GPU programming remains in its infancy. Given this is the target hardware for the proposed algorithm we first discuss the internal memory speeds and limitations of the GPU.

From the CPU, data may be pushed to global, constant and texture memory banks. The speed of this transaction can be limited by either the graphics card's memory bus or the host system's motherboard PCI-E port. The maximum speeds of PCI-E versions 1, 2, and 3, are 4, 8, and 16 GB/s respectively.

The paradigm for parallel GPU computing segments operations first into blocks, and then to threads. As of CUDA 5.0, each block may contain up to 1024 threads which have access to 6 memory types. Global memory or DRAM is the largest bank with 2GB on the consumer grade nVidia GeForce GTX 660 Ti used to evaluate GPU applications in this paper. Global memory is located off chip and resultantly has access latency on the order of 100 times less than on chip shared and register memory. Shared memory is accessible from any thread in the same block but is limited to 48KB. Registers, in contrast, have thread scope and are limited to 63 32-bit registers per thread. For our purposes, local memory can be thought of as low-speed spillover if register memory is filled. Finally, constant and texture memory can be ignored for the proposed implementation purposes.

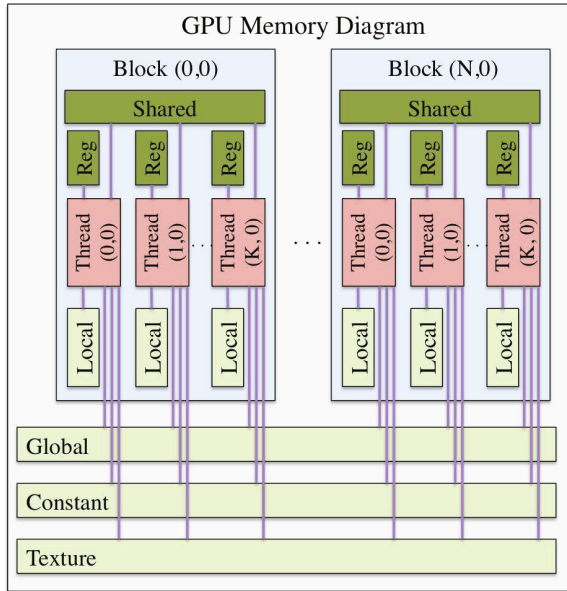


Fig. 1. GPU Memory Diagram

### 3 NSGA II

For multi-objective GAs (MOGA) nondominated sorting genetic algorithm II (NSGA II) is widely considered state-of-the-art [19]. Several key advantages over other popular MOGAs include, reduced computational complexity, elitism, and parameterless fitness sharing operator [5]. The algorithm can be outlined as follows.

1. Random population  $P_0$  is created of size  $N$
2. Evaluate  $P_i$  for all fitness criteria
3. Sort  $P_i$  by nondomination & crowd distance
4. Selection, Recombination, Mutation
5. Resulting population is  $Q_i$  of size  $N$
6.  $R_i = \text{append}(P_i, Q_i)$
7. Sort  $R_i$  by nondomination & partial rank
8.  $P_{i+1} = \text{first } N \text{ in } R_i$
9. Repeat steps 2-8 until exit criteria

where  $N$  is the population size.

Nondomination level is assigned such that no individuals on the same level strictly dominate any other individuals along all fitness axes. In other words, the first level consists of the population's best estimate of the Pareto set, and each subsequent level is the Pareto set of the population excluding all individuals contained by lower levels.

To maximize the distance among found points on the current population Pareto front, each level is internally sorted according to the crowding distance between its nearest neighbors. In this context, crowding distance is computed by sorting the population first on level, and then on each of the fitness function values in ascending order. When the currently sorted fitness function is  $c$ , and level is  $l$  the crowding distance is found from equation 1.

$$\text{Dist}(i) = \frac{1}{2} \sum_{c=1}^2 \sum_{i=\min(l)+1}^{i=\max(l)-1} \frac{P_{l,i+1}.c - P_{l,i-1}.c}{\max(P_l.c) - \min(P_l.c)} \quad (1)$$

where  $P_{(l,i)}.c$  is the value of fitness function  $c$  for the  $i^{\text{th}}$  individual in level  $l$  and  $\max(P_l.c)$  is the maximum fitness value for the  $l^{\text{th}}$  level of population  $P$ .

The individuals with the minimum and maximum fitness values for each level are given an infinite distance. In effect this makes the extremes of each level most likely to be selected for recombination. Even considering NSGA II's diminished computational complexity relative to NSGA it remains impractical for large population sizes. As originally proposed NSGA II is also memory intensive. Namely, in order to reduce the time/computational complexity, storage requirements become  $O(N^2)$  which can further limit population size in memory restricted environments.

## 4 Memory Footprint

In several key areas GPUs are ideally suited to solve genetic algorithms. One primary advantage is the capability to produce random numbers in register memory. The CURAND library does exactly this by allowing each thread to seed its own random number generator, produce, and finally consume random numbers without use of slow-access global memory. This fact, combined with GAs' heavy reliance on random numbers in the selection, recombination, and mutation phases can lead to significant performance gains.

Beyond random number generation, the primary consideration when discussing island based, or decomposed GAs, is memory management. The ideal case is to house all populations in unique blocks of high speed shared memory. This can prove problematic due to the restrictive storage capacity of 48KB. As seen in figure 2, 48KB is a severe limitation for most genome types. Note that computation of the maximum genome length under the heading 'Max G NoVar' contains raw population storage only, while 'Max G Var' incorporates 16 bytes per genome and 32 bytes per population of working variables.

Instances of research which address the memory issue where complex genome types and large population sizes are required most commonly utilize global memory for genome storage. Doing so all but eliminates memory considerations, but access latency suffers dramatically.

In scenarios where fitness functions are sufficiently computationally expensive it may be possible to partially hide memory access lag [14]. When in combination

Population Size				Max G	
bitset	bool	short	int/float	NoVar	Var
128	16	8	4	1500	1483
256	32	16	8	750	733
512	64	32	16	375	358
1024	128	64	32	187	171
2048	256	128	64	93	77
4096	512	256	128	46	30
8192	1024	512	256	23	7
16384	2048	1024	512	11	0
32768	4096	2048	1024	5	0

Fig. 2. Genome Memory Limitations

with parallel sorting and nondomination functions, the GPU can be fully utilized for the majority of genetic algorithm phases.

We propose two technical implementation differences to the usual shared-memory, island-style model which partially alleviate the restricted genome size. First, a bitset is used to enable binary values to be stored at single bit resolution. A side effect when genes are binary is a reduction in memory requests as each 32-bit variable contains 32 genes. For non-binary genome types the bitset can be manipulated to ensure dense packing of any data type to a specified resolution. The second difference is the dynamic ‘as-needed’ use of global memory. Given the knowledge that lowest level and least crowded solutions are most likely to be selected for recombination, these solutions require the most memory accesses. To reduce global memory accesses, the frequently retrieved solutions are housed in shared memory to the maximum extent possible. The remaining less frequently accessed genomes are allocated in global memory.

## 5 NSGA-AD

As previously mentioned, the goal of this NSGA II adaptation is to produce an island-model in which each island or sub-population maps a section of the Pareto optimal frontier. Scalar approaches to decomposition convert MOPs to multiple SOPs by providing weight vectors to each sub-population. The sub-populations’ fitness figures are then evaluated by linearly weighting all fitness values for each individual via the corresponding element of the relevant weight vector. A simple example is shown in figure 3. Note that the appeal this structure is its ease of parallelization on a population granularity. Parallelization of NSGA II is not as straightforward. Applying multiple parallel populations with NSGA II would cause each population map the entire Pareto frontier as in figure 4.

We suggest decomposing the Pareto front similarly to the scalar decomposition approach. Instead of passing a single weight vector, the search space is linearly divided into equal portions by a weight range.

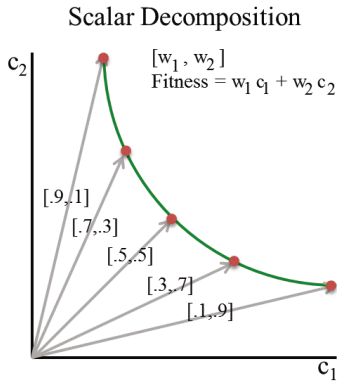


Fig. 3. Scalar Decomposition of Multi-Objective Search Space

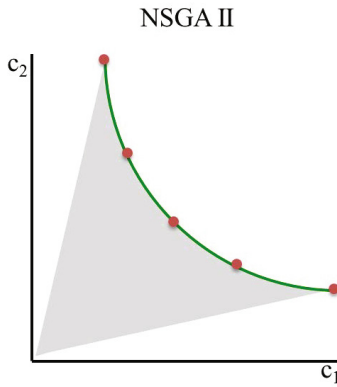


Fig. 4. NSGA II in Multi-Objective Search Space

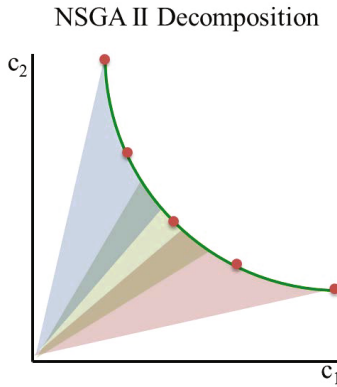


Fig. 5. Proposed Decomposition of Multi-Objective Search Space



To enforce each sub-population tracts to its desired section of the search space, a constraint parameter is introduced. This parameter is such that a solution’s genome is within its constraint when its host sub-population’s range of weight vectors yields the highest value of Fitness in the following.

$$Fitness = \sum_{i=1}^C (w_{d,i} * c_{d,i}) \tag{2}$$

where d is the sub-population index and i the index of each fitness criterion.

Each sub-population has a target area of the C dimensional search space which it is responsible for. This can be seen in figure 5. Once the standard NSGA II sort is complete, each solution’s fitness values are entered into equation 3 with each population’s weights. When the solution’s host population weights yield the highest total fitness value relative to all other weights the solution is said to be within its target space. If any other set of weights yields a higher total value then the solution is outside its target space. Explicitly, the constraint parameter V for each individual in a sub-population is found by:

$$V = \begin{cases} 0 & \sum_{i=1}^C (w_{d,i} * c_{d,i}) > \sum_{i=1}^C (w_{j,i} * c_{j,i}) \\ 1 & otherwise \end{cases} \tag{3}$$

For all sub-populations j where  $j \neq d$ .

The constraint is introduced as selective pressure in each sub-population via an adjustment to the nondomination sort. Any individual which does not satisfy  $V=0$  is decremented a single nondomination level after the initial sort. It would be feasible to treat constraints as additional fitness metrics when performing the NDS. However, this implementation requires a further, computationally expensive, sort.

## 6 Performance: Quality

The test problems selected to review performance statistics were drawn from Zitzlers proposed test problems [20]. From the six suggested problems we select three for use, ZDT1, ZDT2, and ZDT3.

These three examples were chosen as they each exhibit unique properties in Pareto frontier shape. ZDT1 (figure 6) has a convex and continuous Pareto front. ZDT2 (figure 7) is nonconvex and continuous while ZDT3 is convex and discontinuous (figure 8) <sup>1</sup>.

With respect to the quality of produced solutions, we intend only to demonstrate the proposed algorithm yields equivalent results to NSGA II. For this purpose evaluation of convergence and diversity metrics are performed according to the methodology given in [5] to permit direct comparison of originally described performance.

---

<sup>1</sup> Pareto front applicable only between  $x = [0.00000,0.08300], [0.18222,0.25776], [0.40931,0.45388], [0.61839,0.65251], [0.82333,0.85183]$

ZDT1

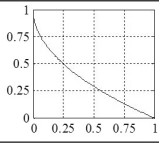
Fitness Functions	Constraints	Plot
$f_1(x) = x_1$ $f_2(x) = g(x) * \left(1 - \sqrt{\frac{x_1}{g(x)}}\right)$ $g(x) = 1 + \frac{9}{n-1} * \sum_{i=2}^G x_i$	$0 \leq x_i \leq 1$ $i = 1, 2, \dots, G$ $G=30$	
Pareto Front		
		$x_2 = 1 - \sqrt{x_1}$

Fig. 6. ZDT1 Test Function

ZDT2

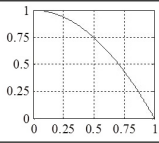
Fitness Functions	Constraints	Plot
$f_1(x) = x_1$ $f_2(x) = g(x) * \left(1 - \left(\frac{x_1}{g(x)}\right)^2\right)$ $g(x) = 1 + \frac{9}{n-1} * \sum_{i=2}^G x_i$	$0 \leq x_i \leq 1$ $i = 1, 2, \dots, G$ $G=30$	
Pareto Front		
		$x_2 = 1 - x_1^2$

Fig. 7. ZDT2 Test Function

ZDT3

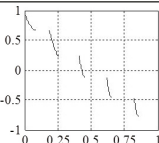
Fitness Functions	Constraints	Plot
$f_1(x) = x_1$ $f_2(x) = g(x) * (1 - \sqrt{x_1/g(x)} - \sin(10\pi x_1))$ $g(x) = 1 + \frac{9}{n-1} * \sum_{i=2}^G x_i$	$0 \leq x_i \leq 1$ $i = 1, 2, \dots, G$ $G=30$	
Pareto Front		
		$x_2 = 1 - \sqrt{x_1} - x_1 \sin(10\pi x_1)$

Fig. 8. ZDT3 Test Function

Convergence quality is determined by selecting 500 points on the known Pareto optimal frontier designated by set H. Minimum Euclidean distance from each individual in the solution set produced by the GA to its nearest counterpart contained in set H is computed. This metric is designated by  $\mathcal{Y}$ .

The second metric of diversity measures the extent to which the solution set is distributed across the Pareto front. Both distribution within the extreme

points of the optimal set and Euclidean distance between the extreme values are considered in accordance with the diversity metric in equation 4.

$$\Delta = \frac{d_f + d_t + \sum_{i=1}^{U(0)-1} |d_i - \bar{d}|}{d_f + d_t + \bar{d}_t (U(0) - 1)} \tag{4}$$

where  $d_f$  and  $d_t$  represent Euclidean distances between maximally distant solutions in the Pareto set and solved for set respectively.

$U(0)$  is the number of nondominated solutions

$\bar{d}$  is taken to be the average Euclidean distance between each nondominated solution and its nearest neighbor notated  $d_i$  along the found frontier.

Mean and variance for both metrics were obtained from 10 independent runs with unique seeds to each threads' random number generator. Parametrization of run limits was set to 25,000 function evaluations per population for both NSGA II and NSGA-AD. Decomposition of the fitness functions in sub-populations of NSGA-AD was fixed such that each of 16 parallel populations received  $(1/14)^{th}$  of the linearly divided fitness space. Note the marginal overlap resulting from this division.

It is necessary to incorporate multiple sub-populations when evaluating performance statistics because a single-island implementation of NSGA-AD is identical to the original NSGA II algorithm. The purpose of the quality test is to determine if dissection of the Pareto front negatively impacts convergence or diversity of solutions. Figure 9 shows the performance of the binary encoded NSGA II and binary encoded NSGA-AD algorithms with each set of test functions.

Algorithm Metric	ZDT1		ZDT2		ZDT3	
	Y	Δ	Y	Δ	Y	Δ
<b>NSGA II Mean</b>	0.000894	0.463292	0.000824	0.435112	0.043411	0.575606
<b>(Binary) Var</b>	0.000000	0.041622	0.000000	0.024607	0.000042	0.005078
<b>NSGA-AD Mean</b>	0.000822	0.386640	0.000826	0.385270	0.006901	0.415940
<b>(Binary) Var</b>	0.000000	0.011585	0.000000	0.009852	0.000000	0.018810

Fig. 9. Performance Metrics

For test problems ZDT1 and ZDT2 the convergence values are lower-bound-limited by incidental distances to the 500 uniformly selected points on the true Pareto front. The values are small enough such that the variance between runs rounds to zero. NSGA-AD demonstrates significantly improved performance relative to NSGA II on the discontinuous ZDT3 fitness function. This can most likely be attributed to the increased ability to speciate when mapping independent populations to subsets of the frontier. We leave exploration of any potential quality gains for a future study and focus primarily on computational speed advantages.

## 7 Performance: Speed

Speed performance of the GPU ported NSGA II variant is measured on a NVIDIA GeForce GTX 660 Ti GPU which contains 1344 computational cores clocked to 980 MHz. For comparison, the same algorithm is tested on an Intel Core i7 950 CPU clocked to 3.06 GHz.

Pop Size	Parallel Sub-Populations				
	1	2	4	8	16
32	0.2	0.3	0.6	1.2	2.2
64	0.3	0.6	1.2	2.4	4.7
128	0.8	1.6	3.2	6.3	12.3
256	2.2	4.4	8.6	17.0	18.6
512	7.1	14.0	27.8	30.8	39.6
1024	27.8	31.0	31.4	31.9	32.4
2048	128.0	128.2	129.2	129.7	129.7

**Fig. 10.** Speedup Results: GPU vs CPU

The speed-ups shown in figure 10 represent the GPU implementation versus a single threaded CPU implementation. All GPU code was written to allow perfect scalability from serial execution to parallelism on the order of 1 thread per solution candidate. Coding for the possibility of serial execution allowed the C code utilized in the CPU implementation to overlap perfectly with its equivalent CUDA code for +95% of the program.

A maximum speed up of 129.7 times was obtained using 16 parallel sub-populations with each sub-population containing 2048 individuals.

## 8 Conclusions

Through use of a GPU we have demonstrated it is possible to attain positive speed-ups for multi-objective genetic algorithms with outputs equivalent to NSGA II. When using the proposed algorithm speed-ups were realized at all tested population sizes with +8 parallel sub-populations. The speed improvement in figure 10 can be seen to level off when the number of parallel sub-populations multiplied by the population size exceeds the number of cores within the GPU. This occurs when the utilization of GPU resources becomes bottlenecked under the described implementation. The proposed algorithm was found to perform at least as well as NSGA II in all test cases along selected quality metrics while generating a maximum of 129.7 times speed up.

## References

1. Akhshabi, M., Haddadnia, J., Akhshabi, M.: Solving flow shop scheduling problem using parallel genetic algorithm. *Procedia Technology* 1, 351–355 (2012)
2. Alba, E., Dorronsoro, B.: Computing nine new best-so-far solutions for Capacitated vrp with cellular Genetic Algorithm. *Information Processing Letters* 98, 225–230 (2006)
3. Alba, E., Troya, J.M.: Analyzing synchronous and asynchronous parallel distributed genetic algorithms. *Future Generation Computer Systems* 17, 451–465 (2001)
4. Davies, R., Clarke, T.: Parallel implementation of a genetic algorithm. *Control Engineering* 3, 11–19 (1995)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6 (2002)
6. Duran, J.P., Kumar, S.A.: CUDA based multi objective parallel genetic algorithms: Adapting evolutionary algorithms for document searches (unpublished)
7. Durillo, J., Nebro, A., Luna, F., Alba, E.: A study of master-slave approaches to parallelize nsga-ii. In: *IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2008*, pp. 1–8 (2008)
8. Gustafson, S., Burke, E.K.: The speciating island model: An alternative parallel evolutionary algorithm. *Journal of Parallel and Distributed Computing* 66, 1025–1036 (2006)
9. Jaros, J.: Multi-gpu island-based genetic algorithm for solving the knapsack problem. *World Congress on Computational Intelligence (June 2012)*
10. Maeda, Y., Ishita, M., Li, Q.: Fuzzy adaptive search method for parallel genetic algorithm with island combination process. *International Journal of Approximate Reasoning* 41, 59–73 (2006)
11. Moreno-Armendariz, M.A., Cruz-Cortes, N., Duchanoy, C.A., Leon-Javier, A., Quintero, R.: Hardware implementation of the elitist compact Genetic Algorithm using Cellular Automata pseudo-random number generator. *Computers and Electrical Engineering* (2013)
12. nVidia: OpenCL Programming Guide for the CUDA Architecture (2009), [http://www.nvidia.com/content/cudazone/download/OpenCL/NVIDIA\\_OpenCL\\_ProgrammingGuide.pdf](http://www.nvidia.com/content/cudazone/download/OpenCL/NVIDIA_OpenCL_ProgrammingGuide.pdf)
13. nVidia: CUDA C Programming Guide (2012), <http://docs.nvidia.com/cuda/cuda-c-programming-guide/>
14. Pospichal, P., Jaros, J.: Gpu-based acceleration of the genetic algorithm, *gECCO Competition* (2009)
15. Rausch, T., Thomas, A., Camp, N.J., Cannon-Albright, L.A., Facelli, J.C.: A parallel genetic algorithm to discover patterns in genetic markers that indicate predisposition to multifactorial disease. *Computers in Biology and Medicine* 28, 826–836 (2008)
16. Solar, M., Parada, V., Urrutia, R.: A parallel genetic algorithm to solve the set-covering problem. *Computers & Operations Research* 29, 1221–1235 (2002)
17. Straßburg, J., Gonzalez-Martel, C., Alexandrov, V.: Parallel genetic algorithms for stock market trading rules. *Procedia Computer Science* 9, 1306–1313 (2012)
18. Tantar, A., Melab, N., Talbi, E.G., Parent, B., Horvath, D.: A parallel hybrid genetic algorithm for protein structure prediction on the computational grid. *Future Generation Computer Systems* 23, 398–409 (2007)
19. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation* 1, 32–49 (2011)
20. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* 8(2), 173–195 (2000)

# Evolutionary Scheduling for Mobile Content Pre-fetching

Omar K. Shoukry and Magda B. Fayek

Cairo University,  
Giza, Egypt  
eng.oks@gmail.com

**Abstract.** Recently, an increasing number of mobile users are eagerly using the cellular network in data applications. In particular, multimedia downloads generated by Internet-capable smart phones and other portable devices (such as tablets) has been widely recognized as the major source for strains in cellular networks, to a degree where service quality for all users is significantly impacted. Lately, patterns in both the content consumption as well as the Wi-Fi access by the users were alleged to be available. In this paper we introduce a technique to schedule the content for prefetching based on mobile usage patterns. This technique utilizes both a content profile as well as a bandwidth profile to schedule content for prefetching. Users can then use the cached version of the content in order to achieve a better user experience and reduce the peak-to-average ratio in mobile networks, especially during peak hours of the day. An experiment using real users traces was conducted and the results after applying the proposed evolutionary scheduling algorithm show that up to 70% of the user content requests can be fulfilled i.e. the content was successfully cached before request.

**Keywords:** Evolutionary algorithms, Genetic algorithms, Content pre-fetching, Mobile users, Behavioral models, Pattern mining, Traffic off-loading.

## 1 Introduction

Recently, an ever increasing demand for the wireless spectrum was witnessed resulting from the exponential growth of mobile data traffic due to the increasing penetration of smart phones and the adoption of bandwidth intensive multimedia applications that cater to diverse users' needs. This trend has been widely recognized as the major cause of cellular networks congestion, forcing leading wireless operators around the world to consider significant additional investments in the cellular infrastructure. This trend is also growing in severity with the increasing number of active smart phone devices, and demand for mobile data will soon globally outpace an existing network capacity. According to [5], data and voice services in North America had similar network loads until May 2007. This ratio has dramatically changed ever since. For instance, it has been reported in March

2009 that the traffic load of data services became nine-fold that of voice services [10]. This, in turn, gives rise to severe network congestion degrading the quality of service (QoS) perceived by mobile users.

Handa [5] suggested three alternative approaches to solve the cellular congestion problem: (i) Scaling, (ii) Optimization and (iii) Network offloading. Although, scaling to 4G/LTE networks may help overcome the congestion problem in the short-term, however this solution remains a temporary measure, at best. Second, network Optimization may help relieve the congestion problem via efficient resource utilization, yet, only to a certain extent. Moreover, this approach is faced with serious issues, e.g., isolation of heavy data users, privacy preservation and policing users' traffic [8]. Finally, offloading to secondary infrastructure provides an alternate path to wireless delivery. According to [11], the majority of traffic (63%) generated by smart phones, tablets and feature phones will transfer onto the fixed network via Wi-Fi by 2015. Since a high percentage of mobile data consumption occurs while indoors, or in motion, data traffic can be offloaded onto complementary fixed networks via Wi-Fi.

This paper embraces a fundamentally different approach that is based on the novel framework of proactive resource allocation pioneered by El Gamal et al. [4]. The basic idea is to exploit the inherent predictability of user behavior to pre-fetch content (i.e. before demand) to relief congested networks and enhance the user experience. There has been ample evidence in the literature pointing to the predictability of mobile user behavior [2]. In [3], the authors argue that a large portion of the Internet users have repetitive content access patterns over time (i.e. days). In addition, mobile users exhibit Wi-Fi access and phone battery patterns that are repetitive over time. Our prime concern in this paper is to characterize and leverage those patterns to retrieve content before demand by intelligent scheduling techniques. This paradigm shift not only enhances the capacity of the cellular network but also results in an enhanced user experience by eliminating the delays and low connection rates that often impair the data connections over the cellular infrastructure.

Scheduling is a classic problem that has been studied in multiple disciplines [9], e.g., processor design, queuing theory, and wireless multiple access. Although the content scheduling problem at hand may be thought of as a classic task scheduling problem, it is inherently multi-objective (delivery rate, content freshness and resource utilization, among other objectives) and the tasks to be scheduled are not necessarily available at the beginning of scheduling. Classic schemes fall short off solving the problem and, hence, calls for novel approaches to address the aforementioned challenges. Multi-objective meta-heuristics are largely used for solving such problems [7], namely multi-objective Evolutionary Algorithms (EA). According to [6], the results obtained have shown that evolutionary algorithms can be effectively applied to the intrinsically multi-objective scheduling problem of large scale space network communications scheduling. Multi-objective flow shop scheduling using Meta-heuristics was discussed by Kumar[7]. The objective therein was to minimize the tardiness, make span, earliness and the number of tardy jobs, concurrently. Genetic algorithms and simulated annealing (SA)

were used to solve this problem, and a hybrid was proposed to provide satisfactory results [13]. Genetic Algorithm (GA) is an evolutionary search heuristic that emulates the process of natural selection to solve complex optimization and search problems. The main challenges of a GA is to find an appropriate genetic representation of the solution domain and selecting an appropriate fitness function for evaluation [12].

Our contribution in this paper is two-fold. First, introduce a novel architecture for the proactive offloading system and its major building blocks, namely the loggers, residing on the mobile phone, the behavioral modelers (profilers) and scheduler, residing on the cloud. Second, the proactive scheduler leveraging the developed stochastic models for the user behavior processes of interest, namely content consumption, Wi-Fi access and phone battery state. Motivated by the sheer complexity of the problem, attributed to its combinatorial nature, we introduce an evolutionary approach that assigns content items to slots without explicitly generating the huge search space of all available solutions to the multi-objective problem. The rest of this paper is organized as follows. Section 2 overviews the overall system architecture. Afterwards, Section 3 formulates and solves the proactive scheduling problem via an Evolutionary algorithm. Section 4 conducts a performance evaluation study with the aid of extensive simulations using real data. Finally, Section 5 concludes the work and point out potential directions for future research.

## 2 System Overview

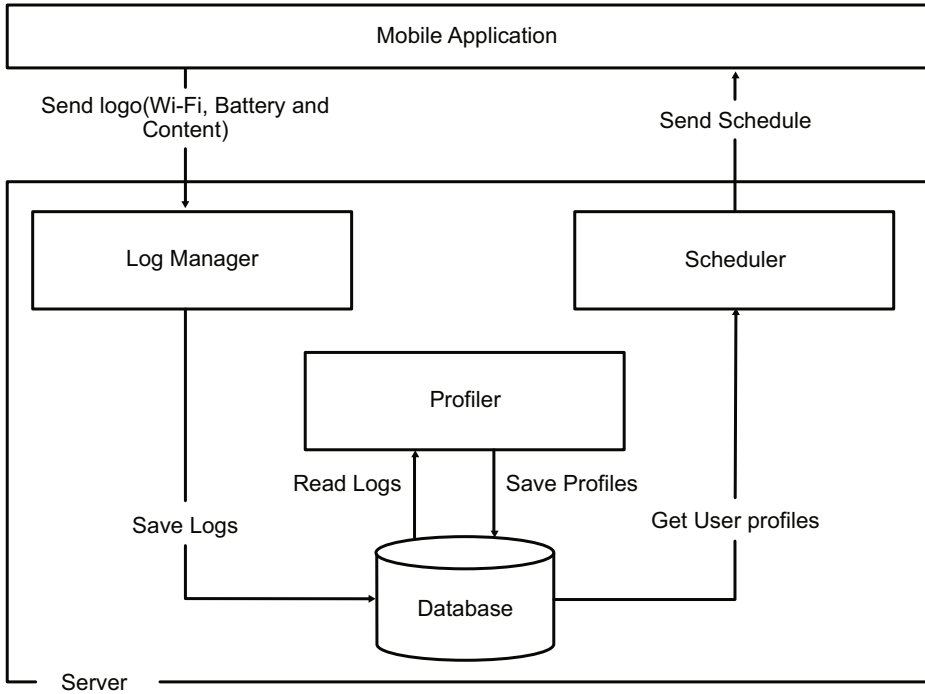
The ultimate objective of the system is to find a schedule for pre-fetching content items, over the course of a day, before actual user demand. The entire system hinges on developing trace-based stochastic models (profiles) for the mobile user behavioral processes of interest, namely content consumption, Wi-Fi and battery state. Given the profiles, an estimate of the average download data rate is computed to estimate the success probability  $SP$  for content caching at a given time slot. Using the calculated schedule, user content can be pre-fetched and made available ahead of demand.

As shown in Fig. 1, the system consists of three major building blocks: (A) User behavior loggers residing on the mobile side, (B) Trace-based stochastic profilers modeling the user behavior and (C) Proactive scheduler which retrieves "predicted" content of interest before demand. The loggers are responsible for capturing the behavior of the mobile user device and sending them to the cloud. The logged data, pertaining to the demand side (content consumption) and resource side (average data rate and battery state), are then used to create representative users' profiles. The scheduler leverages the user profiles to build schedules for content pre-fetching.

### 2.1 Logging User Behavior

This module reside on the user's mobile phone with a responsibility of logging the user Wi-Fi connectivity, the phone battery state and the content usage behavior.





**Fig. 1.** System Architecture

The Connectivity/Data Rate Logger Module is responsible for monitoring the user's Wi-Fi connectivity state and dynamics. Upon each user entry/exit from a Wi-Fi network, the mobile operating system will notify the connectivity logger to append a new log entry including the current connectivity information. The Wi-Fi connectivity model aims at capturing the visited networks, the residence time in each network and the available resources (i.e. bandwidth) throughout each visit. The Battery State Logger Module is responsible for monitoring the battery and listening to the following battery events: Battery charging, Battery low and Battery OK. If one of the previous events occurs, the mobile operating system will notify the battery logger to append a new log entry representing the current battery state and time in the battery log file. Finally, the Content Usage Logger Module is responsible for logging the content consumption of the users' applications traffic.

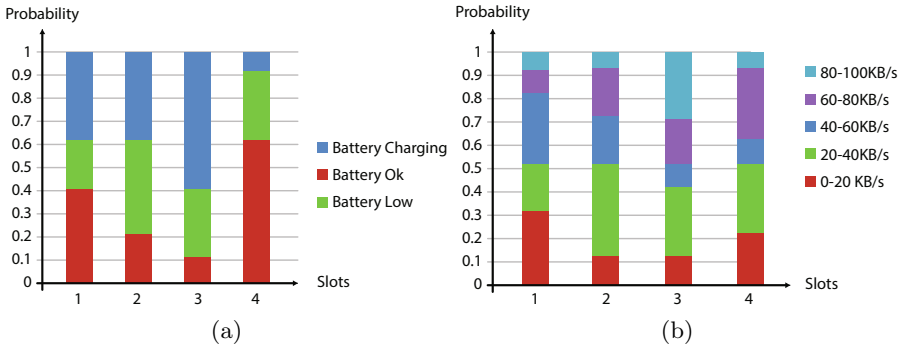
## 2.2 Probabilistic User Behavioral Models (Profiles)

We divide the day into a number of time slots with equal intervals (system parameter). The profiler uses the data from the logs (battery usage, average data rate and the accessed content) to generate representative histograms, which constitute the user profile. The three above mentioned processes generally exhibit

correlation among them. Based on intuition, we assume that the battery state and the available BW processes are correlated. This can be illustrated with the aid of an example. A user on the go, with limited or no available Wi-Fi access, will most probably have a Low Battery state while a charging, or a fully charged battery state, would have higher probability of having a stable Wi-Fi connectivity. This correlation is the prime motivation for introducing a joint Wi-Fi and Battery state model. The joint BW-Battery profiler takes input logs from the Wi-Fi and Battery state to create a representative model for the correlated connectivity and Battery state of the user. In order to simplify the model, the content usage process is assumed to be uncorrelated to the bandwidth and Battery state processes.

The Joint Battery-Download BW Model models the user’s resource availability. As illustrated earlier, the battery logger periodically logs the state of the battery and lists it in terms of the following three states: CHARGING, OK and LOW . On the other hand, the Wi-Fi model provides a breakdown of the probability of a certain Wi-Fi BW rate available during each Wi-Fi logging period. We assume that the available download rate is such that it can be divided into the following five data rate bins in kbps e.g., 0-20, 20-40, 40-60, 60-80, 80-100. Thus, the profiler creates histograms for the battery and bandwidth with 15 different states (3 battery states \* 5 data rate states). For each slot, the probability that the combined state is ("CHARGING and 0-20", "OK and 0-20", "LOW and 0-20", "CHARGING and 20-40" ...etc) is computed from the logs. This probability can be computed as follows:

$$P(State) = \frac{D(State)}{D_{overall}} \tag{1}$$



**Fig. 2.** The (a) Battery profiling histogram, and (b) Wi-Fi profiling histogram

where a joint state is a (Battery, W-Fi BW) pair,  $D(State)$  is the duration that the given (Battery, BW) pair occurs expressed as number of time slots  $X$ , and  $D_{overall}$  is the total amount of time. Examples of the battery and Wi-Fi histogram profiles are shown in Fig. 2.

The Content Usage Profile models the user's content demand. The content usage logger periodically logs the content (application) requests and lists them in terms of the Application Categories. Afterwards, the content usage profiler (modeler) generates a histogram profile for a given user. We assume that the slot has a maximum of ONE request and that we have  $M$  applications categories. Each user accesses  $M$  application categories of interest. The content usage logs are used to determine the  $M$  categories accessed by a given user. Generally, different users may have different values of  $M$ , however,  $M$  is fixed for a given user in a single time slot. The content usage profiler computes the probability of requesting each application category (Content State) within a specific time slot as:

$$P(\text{Content}_{State}) = \frac{F(\text{Content}_{State})}{X} \quad (2)$$

where  $F(\text{Content}_{State})$  is the number of times (frequency) this specific Content State was requested in the past  $X$  similar time slots.

### 3 Proactive Content Scheduler

Using the generated profiles, the scheduler employs the predictable user behavior, and the capabilities of smart phones to meet the ever-growing demand for the limited, non-renewable wireless spectrum. This proactive content scheduler constitutes the center, and most critical, piece of the proposed content delivery system for mobile users. The resulting schedule delivers content before demand (intelligent prefetching) that allows off-peak data offloading and a better user experience regarding content consumption. Intelligent offloading also improves network resource utilization by offloading across time and/or secondary networks (Wi-Fi).

#### 3.1 System Model

The scheduling scope is specified as one day as we assume one segment per day throughout the paper. For any day of the week, a schedule is generated based on the constructed probabilistic model for that day.  $r_i$  and  $b_i$  denote the average download rate and battery state in slot  $i$ , respectively. The system accommodates  $M$  different types of content items (application categories). A day is divided into a number of slots with fixed duration, denoted,  $D$ . Slot duration is the same, and fixed, throughout constructing the behavioral models and scheduling. A single day is split into  $N$  slots where  $N = \frac{24*60}{D(\text{mins})}$ . We assume that a single content item, at maximum, can be retrieved in a single slot. Notice that no content items are retrieved, in a particular slot, due to the lack of connectivity and/or battery resources or violation of content freshness constraints. A scheduling policy, i.e. the slot-content item assignment, for the  $N$  slots is modeled by the vector  $\mathbf{S}$  as follows

$$\mathbf{S} = [C_i^1, C_j^2, \dots, C_k^N]$$

where  $C_i^n$  represents retrieving content item of type  $i$  in slot  $n$  and  $1 \leq i, j, k \leq M$ .

### 3.2 Problem Formulation and Complexity

Our prime objective in this paper is to schedule content item downloads, over the course of a day, in order to maximize the probability of the user having the content cached before demand, subject to data rate, battery and content freshness constraints. Evidently, the optimization problem is combinatorial and, hence, the optimal scheduler has to examine all  $|\mathcal{S}| = (M + 1)^N$  combinations where  $M$  is the number of application categories of interest and  $N$  is the number of slots per day. This, in turn, gives rise to exponential complexity of the optimal policy with the number of slots, which makes the optimal prohibitively complex and practically infeasible. This motivates us to explore the evolutionary proactive scheduler.

### 3.3 Evolutionary GA Scheduler

The proposed evolutionary proactive scheduler hinges on two important notions, namely Reward  $R$  and Utility  $U$  where the Reward captures the benefit of assigning a content item (Application Category) to a given slot and the Utility captures the ability to retrieve this item in the given slot based on the availability of battery and bandwidth resources. The proactive scheduler tries to form a list of tasks with a maximum total utility. The utility of the schedule is the summation of all slots utilities, which is the slot Reward  $R$  multiplied by the success probability  $SP$ . The Reward for each content item in a specific slot depends on the probability of requesting this item within successive slots in a window  $W$ . This leads to assigning the content item to a slot based on the highest probability of being requested in the next time slots. The success probability of the slot depends on the bandwidth and battery state profiled in this slot (available resource). An evolutionary based algorithm is used to generate different solutions and select the results with the maximum total utility. Next, we give a detailed description of the evolutionary proactive scheduler.

First, the scheduling day is divided into a number of slots with constant slot period ( $P$ ). This period is equal to the pre-defined profiling slot length (30 min. in our current implementation). The algorithm schedules tasks, using one slot for each task, for caching relevant content before its demand. After defining the number of slots  $N$  with the beginning and end times of each slot, a random Application Content ( $AC$ ) is assigned to each slot. Each  $AC$  can be one of the  $M$  values (application names). Uniform Crossover is selected for generating subsequent solutions. Randomly exchanging the  $AC$  values of two slots is used as mutation. The reward  $R$  of each slot is calculated and the Utility  $U$  is estimated by multiplying the reward with the success probability. The total schedule fitness is the summation of utilities of all slots. The Reward  $R$  of an  $AC$  at a specific slot ( $t_{Slot}$ ) is defined as

$$R(AC, t_{slot}) = \sum_{t=t_{slot}+1}^W ACProfile[t, AC] \quad (3)$$

---

**Algorithm 1.** Evolutionary GA Scheduler

---

```

READ Content Profiles
READ Battery and Wi-Fi Profiles
i = 0
j = 0
Initialization: Generate Initial random population
while i < N do
    CALCULATE Application Content Reward of each slot
    UPDATE Content Profile
    i = ++
end while
CALCULATE Utility of each slot
Evaluate the fitness of each individual (utility summation) in this population
while j < G do
    Selection : Select best fit individuals (highest utility summation)
    Evolution: Generate new individuals by applying crossover and mutation
    Evaluate the fitness of newly generated individuals
    Replace some of the (least fit) individuals from the old population with new (higher
    fit) individuals
    j = ++
end while

```

---

Thus, the  $R(AC, slot)$  is determined by summing the profiled values (i.e. probabilities) of the selected AC over all slots following the slot of interest and within a window size ( $W$ ) which constitutes a parameter of the system. The window size  $W$  represents the "freshness" duration selected before this cached content becomes obsolete. The whole process is repeated for all slots after modifying the AC profiles values of each scheduled slot. The AC profiles are modified based on the scheduled slot selected AC. The AC profile values are decreased within a window just after the current slot time. This modification reflects the fact that an AC has been scheduled and would be fresh to use for a given window of time.

The Utility  $U$  of a specific slot ( $t_{slot}$ ) having a maximum Reward ( $Max(R)$ ) is given by

$$U(t_{slot}) = SP(t_{slot}) * Max(R) \quad (4)$$

The slot utility represents the benefit from caching this AC at this time slot. The success probability  $SP$  is the probability of successfully caching this AC at this time slot and is defined as the probability of the bandwidth ( $BW$ ) and Battery at slot ( $t_{slot}$ ) allocated by content AC matching the  $BW$  and battery profiles.

After generating the first set of solutions (first generation), crossover and mutation are performed to evolve the current set of solutions. This process is repeated for number of iterations till a final set of solutions is generated. The schedule with the highest fitness is selected as the final solution. The fitness is calculated by summing the utility values of all the slots in the individual. The individual with the highest utility summation is considered the most fit

individual. A final modification is done by combining similar requests (same  $AC$ ) over consecutive slots. This modification adds the number of items of a specific  $AC$  to the first occurrence of consecutive slots and removes duplicates. The final schedule may have more than one item to retrieve in a specific time slot with empty slots to follow.

## 4 Performance Evaluation

A baseline scheduler is implemented to compare the results with the proposed scheduler algorithm. Simulated Annealing (SA) was chosen as a stochastic algorithm for creating random schedules. In the scheduling context, a random list of tasks is generated (assignment of content to time slots). The Utility of the schedule is calculated as shown before using the same Reward notion. The total schedule utility is calculated by summing the utility of all assigned slots. At each iteration a new randomized schedule is generated and selected if it has a higher total utility value than the previous schedule. This is repeated for  $G$  times (stopping condition). The value  $G$  is a fixed value, chosen to accommodate reasonable results in acceptable computation complexity.

### 4.1 Performance Results

Table 1 includes the numerical values of the system parameters used in our performance evaluation study. For all shown results, we assume the segment length is one day. In this section, smart phone traces collected by Rice university LiveLab project are utilized [1] [14]. LiveLab is a methodology to measure real-world smart phone usage and wireless networks with a re-programmable, in-device, logger designed for long-term user studies. LiveLab was deployed for a number of iPhone 3GS users. This includes 24 Rice University students from February 2010 to February 2011, and 10 Houston Community College students from September 2010 to February 2011. While LiveLab logs a variety of measurements, for the sake of system evaluation, application usage data, associated Wi-Fi and data rate, and battery level data were used.

**Table 1.** System Parameters

<i>Parameter</i>	<i>Definition</i>	<i>Value</i>
N	Number os Slots	48
M	Number of Applications	10
P	Slot Period (min)	30
W	Window Size (slots)	6
I	Number of Items per Slot	1
G	Number of generations	1000
L	Number of individuals	100
C	Crossover Probability	0.7
T	Mutation Probability	0.3

It is worth mentioning that some preprocessing on the data to fit the proposed system has been performed. For the application data, user application usage frequency and duration were measured and a set of applications was selected to work with (CNN, ESPN, YouTube, Facebook, Twitter and LastFM). Focusing on this set of application categories is due to their popularity and widespread. For the associated Wi-Fi data, the entries were summarized into a set of Wi-Fi enter/exit entries. For the battery usage data, both the connectivity data and battery level data were used to represent the required battery activities (battery OK, low, and charging).

The GA scheduler and baseline, simulated annealing, SA scheduler, were implemented and tested for 25 users. This set was selected out of the 34 users available from the LiveLab data due to the availability of sufficient Wi-Fi, battery and content logs over a sufficient period of time. Data was divided to a training set of 5 weeks and a testing set. A day was randomly picked from the available traces as the testing day and the previous 5 weeks were used for building the user behavioral models.

**Table 2.** Overall performance of the scheduling algorithm compared to the baseline scheduler for all users

Performance Metric	GA	SA
Hit ratio	53 % $\pm$ 19%	28 % $\pm$ 17%
Cache Utilization	64 % $\pm$ 12%	31% $\pm$ 15%
Time in Cache	90.33 mins $\pm$ 25	121.58 mins $\pm$ 35

Table 2 outlines the performance results for all 25 users. These results show that an average value of 53% hit ratio can be achieved using the implemented GA schedule. A maximum of 83% was achievable, which is higher than the baseline SA algorithm. The average cache utilization was found to be 64% for the GA and 31% for the SA. The time in cache value was measured to be the average time difference (in minutes) between the request time and the time of caching. This indicates how long the item resided in the memory before actually being consumed by the user. An average of an hour and half is a satisfying result given that most of the cached items are videos. This value indicates that caching happens relatively just before consumption and not so soon to keep content as fresh as possible. The results also show that the proposed GA scheduler outperform the SA algorithm w.r.t. the average hit ratio by more than 40%, and yields a higher cache utilization as well as lower time in cache per item.

The proposed scheduling algorithm would intuitively perform better in a resource abundant environment due to the Wi-Fi availability and good battery conditions. To get further insights, users were classified into two main groups: Resource abundant (RA) users, and resource challenged (RC) users. Based on this classification, Table 3 outlines the results using the proposed evolutionary GA scheduler and previously mentioned system parameters.

**Table 3.** Results on different user classifications, resource abundant and resource challenged

Performance Metric	Resource Abundant	Resource Challenged
Hit ratio	74 % $\pm$ 9 %	42 % $\pm$ 7 %
Cache Utilization	90 % $\pm$ 4 %	38% $\pm$ 11%
Time in Cache	94.13 mins $\pm$ 19	72.33 mins $\pm$ 24

The results confirm that the proposed scheduler yields a higher hit ratio (approximately doubled) in an abundant resource environment as opposed to a resource challenged one. Resource abundant users were selected to be users with Wi-Fi availability period of 30% or more during the day and a battery "LOW" state duration less than 50% during the day. The cache utilization is intuitively better since more items are cached in a resource abundant environment. The time in cache, on the other hand, has a relatively lower value in the resource challenged environment due to the fact that fewer number of items are cached (i.e. less competition), so the probability of caching it just before consumption is high.

## 5 Conclusion

This work addresses the congestion problem affecting the current 3G networks caused mostly by an increasingly excessive usage of multimedia applications on smart phones. In this work, a method for pre-fetching content is suggested to reduce the congestion in 3G networks by intelligent offloading. This technique depends on the estimation of different user patterns as requested content, network availability and battery conditions. The proposed algorithm outperforms the baseline simulated annealing by 40%, on the average. In addition, the results confirm that the proposed scheduler can successfully fulfill the users' requests with a high percent (up to 70%) using real-life smart phone traces. This work can be extended along the following directions. First the slot duration can be used as a variable parameter instead of a fixed one. This would expose different users to different modeling and scheduling behavior. Second, some data mining technique can be applied for categorizing users. This categorization would help in recommending extra content for similar users.

## References

1. Shepard, C., Rahmati, A., Tossell, C., Kortum, P.: Livelab: measuring wireless networks and smartphone users in the field. ACM SIGMETRICS Perform (2010)
2. Song, C., et al.: Limits of predictability in human mobility. Science 327, 1018 (2010)
3. Larrabeiti, D., Romeral, R., Azcorra, A., Serrano, P.: Charging for web content pre-fetching in 3g networks. In: 4th International Workshop on Internet Charging and QoS Technology (2004)



4. El Gamal, H., Tadrous, J., Eryilmaz, A.: Proactive resource allocation: Turning predictable behavior into spectral gain. In: 8th Annual Allerton Conference on Communication, Control, and Computing (2010)
5. Handa, A.: Mobile data offload for 3g networks. Intellinet-Technology, Whitepaper (2009)
6. Johnston, M.: An evolutionary algorithm approach to multi-objective scheduling of space network communications. Intelligent Automation and Soft Computing (2008)
7. Kumar, A.: Multi-objective flow shop scheduling using metaheuristics. PHD thesis in Mechanical Engineering (2011)
8. Lee, K., Rhee, I.: Mobile data offloading: How much can wifi deliver. In: 10th Annual SIGCOMM Conference on Computer and Data Communication Networks (2010)
9. Pinedo, M.: Scheduling: Theory, algorithms and systems. Springer, New York (2012)
10. Cisco Technical Report: Report, Cisco Visual Networking Index (February 2012), <http://www.cisco.com>
11. Research, J.: Relief ahead for mobile data networks as 63% of traffic to move onto fixed networks via wifi and femtocells by 2015 (April 2011), <http://juniperresearch.com/viewpressrelease.php?pr=240>
12. Salomon, R.: Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. a survey of some theoretical and practical aspects of genetic algorithms. Biosystems (1996)
13. Suman, B., Kumar, P.: A survey of simulated annealing as a tool for single and multiobjective optimization. Journal of the Operational Research Society (2006)
14. Wang, Z., Xiaozhu Lin, F., Zhong, L., Chishtie, M.: How far can client-only solutions go for mobile browser speed? In: Int. World Wide Web Conf. (2012)

# Author Index

- Alghamdi, Mansour 57  
Alkanhal, Mohamed 57
- Ballet, Pascal 96  
Berrocal-Plaza, Víctor 19  
Birattari, Mauro 121
- Cotta, Carlos 205  
Czeizler, Eugen 31
- D'Angelo, Massimiliano 45
- Eiben, A.E. 45  
Elhadj, Yahya O. Mohamed 57  
Elouedi, Zied 80
- Fayek, Magda B. 228  
Fernandes, Carlos M. 205
- Gautam, Vinay Kumar 68  
Gomez-Pulido, Juan A. 145
- Haddow, Pauline C. 68  
Hentech, Rim 80
- Jeannin-Girardon, Anne 96  
Jenhani, Ilyes 80  
Jiménez Laredo, Juan Luis 205  
Jin, Lingling 108
- Khaluf, Yara 121  
Kordos, Mirosław 133  
Kuiper, Martin 68
- Lanza-Gutiérrez, Jose M. 145  
Lehman, Joel 1
- Martí Campoy, Antonio 157  
McQuillan, Ian 108  
Merelo, Juan Julián 205  
Miikkulainen, Risto 1  
Mizuki, Takaaki 193
- Nagy, Marius 169, 181  
Nagy, Naya 169, 181  
Nishida, Takuya 193  
Nogueras, Rafael 205  
Nyman, Rickard 217
- Orponen, Pekka 31  
Ors Carot, Rafael 157
- Rammig, Franz 121  
Rice, Oliver 217  
Rodin, Vincent 96  
Rodríguez-Ballester, Francisco 157  
Rosa, Agostinho C. 205  
Rusiecki, Andrzej 133
- Sánchez-Pérez, Juan M. 19  
Shoukry, Omar K. 228  
Smith, Robert E. 217  
Sone, Hideaki 193
- Vega-Rodríguez, Miguel A. 19, 145
- Weel, Berend 45