Cristian S. Calude
Michael J. Dinneen
Gheorghe Păun
Mario J. Pérez-Jiménez
Grzegorz Rozenberg (Eds.)

LNCS 3699

Unconventional Computation

4th International Conference, UC 2005
Sevilla, Spain, October 2005
Proceedings

Springer

Cristian S. Calude Michael J. Dinneen
Gheorghe Păun Mario J. Pérez-Jiménez
Grzegorz Rozenberg (Eds.)

Unconventional Computation

4th International Conference, UC 2005
Sevilla, Spain, October 3 – 7, 2005
Proceedings

Springer

Volume Editors

Cristian S. Calude
Michael J. Dinneen
University of Auckland, Department of Computer Science, Private Bag 92019, Auckland, New Zealand
E-mail: {cristian, mjd}@cs.auckland.ac.nz

Gheorghe Păun
Mario J. Pérez-Jiménez
Sevilla University, Department of Computer Science and Artificial Intelligence, Spain
E-mail: {gpaun, marper}@us.es

Grzegorz Rozenberg
Leiden University
Leiden Center for Natural Computing
Leiden, The Netherlands
E-mail: grozenberg@gmail.com

Library of Congress Control Number: 2005932252

CR Subject Classification (1998): F.1, F.2

ISSN 0302-9743
ISBN-10 3-540-29100-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-29100-8 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11560319 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3699

Preface

The Fourth International Conference on **Unconventional Computation, UC 2005**, organized under the auspices of EATCS by the Centre for Discrete Mathematics and Theoretical Computer Science and the Department of Computer Science and Artificial Intelligence of the University of Seville, was held in Seville, October 3–7, 2005.

Seville, one of the most beautiful cities in Spain, is at its best in October. An explosion of colour and contrast: flamenco, bullfighting, and a lively atmosphere in the streets due to the open and friendly nature of its people. The river Guadalquivir, the Cathedral and the Golden Tower are all places full of magic where the visitor can feel the spirit of a city which is eternally romantic.

The series of International Conferences **Unconventional Computation (UC)**, `https://www.cs.auckland.ac.nz/CDMTCS/conferences/uc/` is devoted to all aspects of unconventional computation, theory as well as experiments and applications. Typical, but not exclusive, topics are: natural computing including quantum, cellular, molecular, neural and evolutionary computing; chaos and dynamical systems based computing; and various proposals for computations that go beyond the Turing model.

The first venue of the Unconventional Computation Conference (formerly called Unconventional Models of Computation) was Auckland, New Zealand in 1998; subsequent sites of the conference were Brussels, Belgium in 2000 and Kobe, Japan in 2002.

The titles of the proceedings volumes from past UC Conferences are as follows:

1. C.S. Calude, J. Casti, M.J. Dinneen (eds.). *Unconventional Models of Computation*, Springer-Verlag, Singapore, 1998, viii + 426 pp. ISBN: 981-3083-69-7.
2. I. Antoniou, C.S. Calude, M.J. Dinneen (eds.). *Unconventional Models of Computation, UMC'2K*, Springer-Verlag, London, December 2000, xi + 301 pp. ISBN 1-85233-417-0.
3. C.S. Calude, M.J. Dinneen, F. Peper (eds.). *Third International Conference, UMC 2002*, Lecture Notes in Computer Science, Vol. 2509, Springer-Verlag, Heidelberg, 2002, vii + 331 pp. ISBN: 3-540-44311-8.

The Steering Committee of the series of International Conferences **Unconventional Computation** includes T. Bäck (Leiden, The Netherlands), C.S. Calude (Auckland, NZ, co-chair), L.K. Grover (Murray Hill, NJ, USA), J. van Leeuwen (Utrecht, The Netherlands), S. Lloyd (Cambridge, MA, USA), Gh. Păun (Seville, Spain and Bucharest, Romania), T. Toffoli (Boston, MA, The five invited speakers of the conference were:

T. Bäck (Leiden, The Netherlands): *Using genetic algorithms to evolve behaviour in cellular automata*
L. Grover (Murray Hill, USA): *Quantum searching amidst uncertainty*
S. Istrail (Providence, USA): *Logic functions of the genomic cis-regulatory code*
N. Seeman (New York, USA): *Structural DNA nanotechnology: Molecular constructions and computations*
C. Torras (Barcelona, Spain): *Natural inspiration for artificial adaptivity: Some neurocomputing experiences in robotics*

UC 2005 included the following three tutorials:

S. Istrail (Providence, USA): *Logic of networks*
I. Petre (Turku, Finland) and G. Rozenberg (Leiden, The Netherland): *Computing with living cells*
Gh. Păun (Seville, Spain): *Elementary aspects of membrane computing*

The Programme Committee thanks the much appreciated work done by the paper reviewers for the conference. These experts were:

S. Basu	D. Karig	M. Margenstern	M. A. Stay
H.-G. Beyer	T. D. Kieu	G. Mauri	K. Svozil
M. Burgin	S. Kobayashi	K. Morita	H. Umeo
C. S. Calude	O. H. Ibarra	M. Mozer	H. T. Wareham
G. Csardi	V. V. Ivanov	B. Ömer	R. Weiss
M. J. Dinneen	N. Jonoska	Gh. Păun	T. Yokomori
P. Érdi	N. Krasnogor	M. J. Pérez-Jiménez	
A. Ekert	J. van Leeuwen	I. Petre	
M. P. Frank	N. Lucas	U. Speidel	

The Programme Committee consisting of L. Accardi (Rome, Italy), H.-G. Beyer (Dornbirn, Austria), M. Burgin (Los Angeles, USA) C. S. Calude (chair; Auckland, NZ), M. J. Dinneen (secretary; Auckland, NZ), P. Érdi (Kalamazoo, USA), A. Ekert (Cambridge, UK), M. P. Frank (Tallahassee, USA), V. V. Ivanov (Dubna, Russia), N. Jonoska (Tampa, USA), N. Krasnogor (Nottingham, UK), J. van Leeuwen (Utrecht, The Netherlands), K. Morita (Hiroshima, Japan), M. Mozer (Boulder, USA), Gh. Păun (Seville, Spain), I. Petre, (Turku, Finland), H. Umeo (Osaka, Japan), R. Weiss (Princeton, USA), T. Yokomori (Tokyo, Japan), selected 18 papers (out of 29) to be presented as regular contributions

We extend our thanks to all members of the Conference Committee, particularly to M. Cavaliere, C. Graciani Díaz, M. A. Gutiérrez Naranjo, A. Nepomuceno Fernández, Gh. Păun, M. J. Pérez Jiménez (chair), F. J. Romero Campero, A. Riscos Núñez, A. Romero Jiménez, F. Sancho Caparrini, D. Sburlan, and U. Speidel (registration), for their invaluable organizational work.

We thank the University of Seville and the Centre for Discrete Mathematics of the University of Auckland for their technical support. The hospitality of our hosts, the Department of Computer Science and Artificial Intelligence of the University of Seville, is much appreciated. The conference was partially supported by the project TIN2004-23021-E of the Ministerio de Educación y Ciencia of Spain, by the II Plan Propio of the University of Seville, and by the Acción Coordinada IMUS 2003 of the Junta de Andalucía: we extend to all our gratitude.

It is a great pleasure to acknowledge the fine co-operation with the Lecture Notes in Computer Science team of Springer for producing this volume in time for the conference.

July 2005

C. S. Calude
M. J. Dinneen
Gh. Păun
M. J. Pérez Jiménez
G. Rozenberg

Lecture Notes in Computer Science

For information about Vols. 1–3624

please contact your bookseller or Springer

Vol. 3728: V. Paliouras, J. Vounckx, D. Verkest (Eds.), Integrated Circuit and System Design. XV, 753 pages. 2005.

Vol. 3726: L.T. Yang, O.F. Rana, B. Di Martino, J. Dongarra (Eds.), High Performance Computing and Communcations. XXVI, 1116 pages. 2005.

Vol. 3718: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing. XII, 502 pages. 2005.

Vol. 3717: B. Gramlich (Ed.), Frontiers of Combining Systems. X, 321 pages. 2005. (Subseries LNAI).

Vol. 3715: E. Dawson, S. Vaudenay (Eds.), Progress in Cryptology – Mycrypt 2005. XI, 329 pages. 2005.

Vol. 3714: H. Obbink, K. Pohl (Eds.), Software Product Lines. XIII, 235 pages. 2005.

Vol. 3713: L. Briand, C. Williams (Eds.), Model Driven Engineering Languages and Systems. XV, 722 pages. 2005.

Vol. 3712: R. Reussner, J. Mayer, J.A. Stafford, S. Overhage, S. Becker, P.J. Schroeder (Eds.), Quality of Software Architectures and Software Quality. XIII, 289 pages. 2005.

Vol. 3711: F. Kishino, Y. Kitamura, H. Kato, N. Nagata (Eds.), Entertainment Computing - ICEC 2005. XXIV, 540 pages. 2005.

Vol. 3710: M. Barni, I. Cox, T. Kalker, H.J. Kim (Eds.), Digital Watermarking. XII, 485 pages. 2005.

Vol. 3708: J. Blanc-Talon, W. Philips, D. Popescu, P. Scheunders (Eds.), Advanced Concepts for Intelligent Vision Systems. XXII, 725 pages. 2005.

Vol. 3706: H. Fuks, S. Lukosch, A.C. Salgado (Eds.), Groupware: Design, Implementation, and Use. XII, 378 pages. 2005.

Vol. 3703: F. Fages, S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning. VIII, 163 pages. 2005.

Vol. 3702: B. Beckert (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. XIII, 343 pages. 2005. (Subseries LNAI).

Vol. 3699: C.S. Calude, M.J. Dinneen, G. Păun, M. J. Pérez-Jiménez, G. Rozenberg (Eds.), Unconventional Computation. XI, 267 pages. 2005.

Vol. 3698: U. Furbach (Ed.), KI 2005: Advances in Artificial Intelligence. XIII, 409 pages. 2005. (Subseries LNAI).

Vol. 3697: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005, Part II. XXXII, 1045 pages. 2005.

Vol. 3696: W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny (Eds.), Artificial Neural Networks: Biological Inspirations – ICANN 2005, Part I. XXXI, 703 pages. 2005.

Vol. 3695: M.R. Berthold, R. Glen, K. Diederichs, O. Kohlbacher, I. Fischer (Eds.), Computational Life Sciences. XI, 277 pages. 2005. (Subseries LNBI).

Vol. 3694: M. Malek, E. Nett, N. Suri (Eds.), Service Availability. VIII, 213 pages. 2005.

Vol. 3693: A.G. Cohn, D.M. Mark (Eds.), Spatial Information Theory. XII, 493 pages. 2005.

Vol. 3692: R. Casadio, G. Myers (Eds.), Algorithms in Bioinformatic. X, 436 pages. 2005. (Subseries LNBI).

Vol. 3691: A. Gagalowicz, W. Philips (Eds.), Computer Analysis of Images and Patterns. XIX, 865 pages. 2005.

Vol. 3690: M. Pěchouček, P. Petta, L.Z. Varga (Eds.), Multi-Agent Systems and Applications IV. XVII, 667 pages. 2005. (Subseries LNAI).

Vol. 3687: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Image Analysis, Part II. XXV, 809 pages. 2005.

Vol. 3686: S. Singh, M. Singh, C. Apte, P. Perner (Eds.), Pattern Recognition and Data Mining, Part I. XXVI, 689 pages. 2005.

Vol. 3685: V. Gorodetsky, I. Kotenko, V. Skormin (Eds.), Computer Network Security. XIV, 480 pages. 2005.

Vol. 3684: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part IV. LXXIX, 933 pages. 2005. (Subseries LNAI).

Vol. 3683: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. LXXX, 1397 pages. 2005. (Subseries LNAI).

Vol. 3682: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. LXXIX, 1371 pages. 2005. (Subseries LNAI).

Vol. 3681: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LXXX, 1319 pages. 2005. (Subseries LNAI).

Vol. 3679: S.d.C. di Vimercati, P. Syverson, D. Gollmann (Eds.), Computer Security – ESORICS 2005. XI, 509 pages. 2005.

Vol. 3678: A. McLysaght, D.H. Huson (Eds.), Comparative Genomics. VIII, 167 pages. 2005. (Subseries LNBI).

Vol. 3677: J. Dittmann, S. Katzenbeisser, A. Uhl (Eds.), Communications and Multimedia Security. XIII, 360 pages. 2005.

Vol. 3676: R. Glück, M. Lowry (Eds.), Generative Programming and Component Engineering. XI, 448 pages. 2005.

Vol. 3675: Y. Luo (Ed.), Cooperative Design, Visualization, and Engineering. XI, 264 pages. 2005.

Vol. 3674: W. Jonker, M. Petković (Eds.), Secure Data Management. X, 241 pages. 2005.

Vol. 3673: S. Bandini, S. Manzoni (Eds.), AI*IA 2005: Advances in Artificial Intelligence. XIV, 614 pages. 2005. (Subseries LNAI).

Vol. 3672: C. Hankin, I. Siveroni (Eds.), Static Analysis. X, 369 pages. 2005.

Vol. 3671: S. Bressan, S. Ceri, E. Hunt, Z.G. Ives, Z. Bellahsène, M. Rys, R. Unland (Eds.), Database and XML Technologies. X, 239 pages. 2005.

Vol. 3670: M. Bravetti, L. Kloul, G. Zavattaro (Eds.), Formal Techniques for Computer Systems and Business Processes. XIII, 349 pages. 2005.

Vol. 3666: B.D. Martino, D. Kranzlmüller, J. Dongarra (Eds.), Recent Advances in Parallel Virtual Machine and Message Passing Interface. XVII, 546 pages. 2005.

Vol. 3665: K. S. Candan, A. Celentano (Eds.), Advances in Multimedia Information Systems. X, 221 pages. 2005.

Vol. 3664: C. Türker, M. Agosti, H.-J. Schek (Eds.), Peer-to-Peer, Grid, and Service-Orientation in Digital Library Architectures. X, 261 pages. 2005.

Vol. 3663: W.G. Kropatsch, R. Sablatnig, A. Hanbury (Eds.), Pattern Recognition. XIV, 512 pages. 2005.

Vol. 3662: C. Baral, G. Greco, N. Leone, G. Terracina (Eds.), Logic Programming and Nonmonotonic Reasoning. XIII, 454 pages. 2005. (Subseries LNAI).

Vol. 3661: T. Panayiotopoulos, J. Gratch, R. Aylett, D. Ballin, P. Olivier, T. Rist (Eds.), Intelligent Virtual Agents. XIII, 506 pages. 2005. (Subseries LNAI).

Vol. 3660: M. Beigl, S. Intille, J. Rekimoto, H. Tokuda (Eds.), UbiComp 2005: Ubiquitous Computing. XVII, 394 pages. 2005.

Vol. 3659: J.R. Rao, B. Sunar (Eds.), Cryptographic Hardware and Embedded Systems – CHES 2005. XIV, 458 pages. 2005.

Vol. 3658: V. Matoušek, P. Mautner, T. Pavelka (Eds.), Text, Speech and Dialogue. XV, 460 pages. 2005. (Subseries LNAI).

Vol. 3657: F.S. de Boer, M.M. Bonsangue, S. Graf, W.-P. de Roever (Eds.), Formal Methods for Components and Objects. VIII, 325 pages. 2005.

Vol. 3656: M. Kamel, A. Campilho (Eds.), Image Analysis and Recognition. XXIV, 1279 pages. 2005.

Vol. 3655: A. Aldini, R. Gorrieri, F. Martinelli (Eds.), Foundations of Security Analysis and Design III. VII, 273 pages. 2005.

Vol. 3654: S. Jajodia, D. Wijesekera (Eds.), Data and Applications Security XIX. X, 353 pages. 2005.

Vol. 3653: M. Abadi, L. de Alfaro (Eds.), CONCUR 2005 – Concurrency Theory. XIV, 578 pages. 2005.

Vol. 3652: A. Rauber, S. Christodoulakis, A M. Tjoa (Eds.), Research and Advanced Technology for Digital Libraries. XVIII, 545 pages. 2005.

Vol. 3650: J. Zhou, J. Lopez, R.H. Deng, F. Bao (Eds.), Information Security. XII, 516 pages. 2005.

Vol. 3649: W.M. P. van der Aalst, B. Benatallah, F. Casati, F. Curbera (Eds.), Business Process Management. XII, 472 pages. 2005.

Vol. 3648: J.C. Cunha, P.D. Medeiros (Eds.), Euro-Par 2005 Parallel Processing. XXXVI, 1299 pages. 2005.

Vol. 3646: A. F. Famili, J.N. Kok, J.M. Peña, A. Siebes, A. Feelders (Eds.), Advances in Intelligent Data Analysis VI. XIV, 522 pages. 2005.

Vol. 3645: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part II. XIII, 1010 pages. 2005.

Vol. 3644: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Part I. XXVII, 1101 pages. 2005.

Vol. 3643: R. Moreno Díaz, F. Pichler, A. Quesada Arencibia (Eds.), Computer Aided Systems Theory – EUROCAST 2005. XIV, 629 pages. 2005.

Vol. 3642: D. Ślezak, J. Yao, J.F. Peters, W. Ziarko, X. Hu (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part II. XXIII, 738 pages. 2005. (Subseries LNAI).

Vol. 3641: D. Ślezak, G. Wang, M. Szczuka, I. Düntsch, Y. Yao (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part I. XXIV, 742 pages. 2005. (Subseries LNAI).

Vol. 3639: P. Godefroid (Ed.), Model Checking Software. XI, 289 pages. 2005.

Vol. 3638: A. Butz, B. Fisher, A. Krüger, P. Olivier (Eds.), Smart Graphics. XI, 269 pages. 2005.

Vol. 3637: J. M. Moreno, J. Madrenas, J. Cosp (Eds.), Evolvable Systems: From Biology to Hardware. XI, 227 pages. 2005.

Vol. 3636: M.J. Blesa, C. Blum, A. Roli, M. Sampels (Eds.), Hybrid Metaheuristics. XII, 155 pages. 2005.

Vol. 3634: L. Ong (Ed.), Computer Science Logic. XI, 567 pages. 2005.

Vol. 3633: C. Bauzer Medeiros, M. Egenhofer, E. Bertino (Eds.), Advances in Spatial and Temporal Databases. XIII, 433 pages. 2005.

Vol. 3632: R. Nieuwenhuis (Ed.), Automated Deduction – CADE-20. XIII, 459 pages. 2005. (Subseries LNAI).

Vol. 3631: J. Eder, H.-M. Haav, A. Kalja, J. Penjam (Eds.), Advances in Databases and Information Systems. XIII, 393 pages. 2005.

Vol. 3630: M.S. Capcarrere, A.A. Freitas, P.J. Bentley, C.G. Johnson, J. Timmis (Eds.), Advances in Artificial Life. XIX, 949 pages. 2005. (Subseries LNAI).

Vol. 3629: J.L. Fiadeiro, N. Harman, M. Roggenbach, J. Rutten (Eds.), Algebra and Coalgebra in Computer Science. XI, 457 pages. 2005.

Vol. 3628: T. Gschwind, U. Aßmann, O. Nierstrasz (Eds.), Software Composition. X, 199 pages. 2005.

Vol. 3627: C. Jacob, M.L. Pilat, P.J. Bentley, J. Timmis (Eds.), Artificial Immune Systems. XII, 500 pages. 2005.

Vol. 3626: B. Ganter, G. Stumme, R. Wille (Eds.), Formal Concept Analysis. X, 349 pages. 2005. (Subseries LNAI).

Vol. 3625: S. Kramer, B. Pfahringer (Eds.), Inductive Logic Programming. XIII, 427 pages. 2005. (Subseries LNAI).

Table of Contents

Invited Papers

Regular Papers

Using Genetic Algorithms to Evolve Behavior in Cellular Automata

Thomas Bäck[1,2] and Ron Breukelaar[1,*]

[1] Universiteit Leiden, LIACS, P.O. Box 9512, 2300 RA Leiden, The Netherlands
{baeck, rbreukel}@liacs.nl
[2] NuTech Solutions GmbH, Martin Schmeißer Weg 15, 44227 Dortmund, Germany
{baeck}@nutechsolutions.de

Abstract. It is an unconventional computation approach to evolve solutions instead of calculating them. Although using evolutionary computation in computer science dates back to the 1960s, using an evolutionary approach to program other algorithms is not that well known. In this paper a genetic algorithm is used to evolve behavior in cellular automata. It shows how this approach works for different topologies and neighborhood shapes. Some different one dimensional neighborhood shapes are investigated with the genetic algorithm and yield surprisingly good results.

1 Introduction

Evolutionary Algorithms is the name for the algorithms in the field of Evolutionary Computation which is a subfield of Natural Computing and already exists more than 40 years. It was born from the idea to use principles of natural evolution as a paradigm for solving search and optimization problem in high-dimensional combinatorial or continuous search spaces. The most widely known instances are genetic algorithms [9,10,11], genetic programming [12,13], evolution strategies [16,17,18,19], and evolutionary programming [7,6]. A detailed introduction to all these algorithms can be found e.g. in the Handbook of Evolutionary Computation [1].

Today the Evolutionary Computation field is very active. It involves fundamental research as well as a variety of applications in areas ranging from data analysis and machine learning to business processes, logistics and scheduling, technical engineering, and others. Across all these fields, evolutionary algorithms have convinced practitioners by the results obtained on hard problems that they are very powerful algorithms for such applications. The general working principle of all instances of evolutionary algorithms is based on a program loop that involves simplified implementations of the operators mutation, recombination, selection, and fitness evaluation on a set of candidate solutions (often called a

* Part of the research was funded by the Foundation for Fundamental Research on Matter (FOM), Utrecht, The Netherlands, *project: "An evolutionary approach to many-parameter physics"*.

C.S. Calude et al. (Eds.): UC 2005, LNCS 3699, pp. 1–10, 2005.

population of individuals) for a given problem. In this general setting, mutation corresponds to a modification of a single candidate solution, typically with a preference for small variations over large variations. Recombination corresponds to an exchange of components between two or more candidate solutions. Selection drives the evolutionary process towards populations of increasing average fitness by preferring better candidate solutions to proliferate with higher probability to the next generation than worse candidate solutions. By fitness evaluation, the calculation of a measure of goodness associated with candidate solutions is meant, i.e., the fitness function corresponds to the objective function of the optimization problem at hand.

No attempt will be made to give a complete introduction or overview of evolutionary algorithms, as there are many good introductory books on the topic available, e.g. [1], instead an example of how to use evolutionary algorithms to parameterize other algorithms will be given.

Evolving parameters for complex algorithms could also be viewed as an inverse design problem, i.e. a problem where the target design (behavior of the algorithm to be parameterized) is known, but the way to achieve this is unknown. The inverse design of cellular automata (CA) is such a problem. Cellular automata are used in many fields to generate a global behavior with local rules. Finding the rules that display a desired behavior can be a hard task especially when it comes to real world problems. This paper uses a genetic algorithm to generate the transition rules for cellular automata, thus evolving global behavior with local rules using a genetic base.

2 Cellular Automata

According to [20] cellular automata (CA) are mathematical idealizations of physical systems in which space and time are discrete, and physical quantities take on a finite set of discrete values. The simplest CA is one dimensional and can be viewed as an array of ones and zeros of width N, where the first position of the array is linked to the last position. In other words, defining a row of positions $C = \{a_1, a_2, ..., a_N\}$ where C is a CA of width N and a_N is adjacent to a_1.

The neighborhood s_n of a_n is defined as the local set of positions with a distance to a_n along the connected chain which is no more than a certain radius (r). This for instance means that $s_2 = \{a_{148}, a_{149}, a_1, a_2, a_3, a_4, a_5\}$ for $r = 3$ and $N = 149$. Note that for one dimensional CA the size of the neighborhood is always equal to $2r + 1$.

The values in a CA can be altered all at the same time (synchronous) or at different times (asynchronous). Only synchronous CA are considered in this paper. In the synchronous approach at every time step (t) every cell state in the CA is recalculated according to the states of the neighborhood using a certain transition rule $\Theta : \{0,1\}^{2r+1} \rightarrow \{0,1\}, s_i \rightarrow \Theta(s_i)$. This rule basically is a one-to-one mapping that defines an output value for every possible set of input values, the input values being the 'state' of a neighborhood. The state of a_n at time t is written as a_n^t, the state of s_n at time t as s_n^t and the state of the entire CA C

at time t as C^t so that C^0 is the initial state and $\forall n = 1, \ldots, N$ $a_n^{t+1} = \Theta(s_n^t)$. Given $C^t = \{a_1^t, ..., a_N^t\}$, C^{t+1} can be defined as $\{\Theta(s_1^t), ..., \Theta(s_N^t)\}$.

Because $a_n \in \{0, 1\}$ the number of possible states of s_n equals 2^{2r+1}. Because all possible binary representations of m where $0 \leq m < 2^{2r+1}$ can be mapped to a unique state of the neighborhood, Θ can be written as a row of ones and zeros $R = \{b_1, b_2, ..., b_{2^{2r+1}}\}$ where b_m is the output value of the rule for the input state that maps to the binary representation of $m - 1$. A rule therefore has a length that equals 2^{2r+1} and so there are $2^{2^{2r+1}}$ possible rules for a binary one dimensional CA. This is a huge number of possible rules (if $r = 3$ this sums up to about $3,4 \times 10^{28}$) each with a different behavior.

One of the interesting things about these and other CA is that certain rules tend to exhibit organizational behavior, independently of the initial state of the CA. This behavior also demonstrates there is some form of communication going on in the CA over longer distances than the neighborhood allows directly. In [14] the authors examine if these simple CA are able to perform tasks that need positions in a CA to work together and use some form of communication. One problem where such a communication seems required in order to give a good answer is the Majority Problem (as described in section 3). A genetic algorithm is used to evolve rules for one dimensional CA that do a good job of solving the Majority Problem [14] and it is shown how these rules seem to send "particles" and communicate by using these particles [15]. These results imply that even very simple cells in one dimensional cellular automata can communicate and work together to form more complex and powerful behavior.

Previous work [4] suggested that using multi dimensional CA works a lot better than using one dimensional CA. This paper will try to shed light on what effect a different topology or neighborhood shape can have, and a measure will be given to classify these different CA.

3 Majority Problem

One of the best known global problems that is (partly) solvable with local rules if the Majority Problem. The Majority Problem can be defined as follows:

Given a set $A = \{a_1, ..., a_n\}$ with n odd and $a_m \in \{0, 1\}$ for all $1 \leq m \leq n$, answer the question: 'Are there more ones than zeros in A?'.

The Majority Problem first does not seem to be a very difficult problem to solve. It seems only a matter of counting the ones in the set and then comparing them to the number of zeros. Yet when this problem has to be solved within the framework of a CA it becomes a lot more difficult. This is because the rule in a CA does not let a position look past its neighborhood and that is why the cells all have to work together and use some form of communication.

Given that the relative number of ones in C^0 is written as λ, in a simple binary CA the Majority Problem can be defined as:

Find a transition rule that, given an initial state of a CA with N odd and a finite number of iterations to run (I), will result in an 'all zero' state if $\lambda < 0.5$ and an 'all one' state otherwise. The 'all zero' state being the state in which

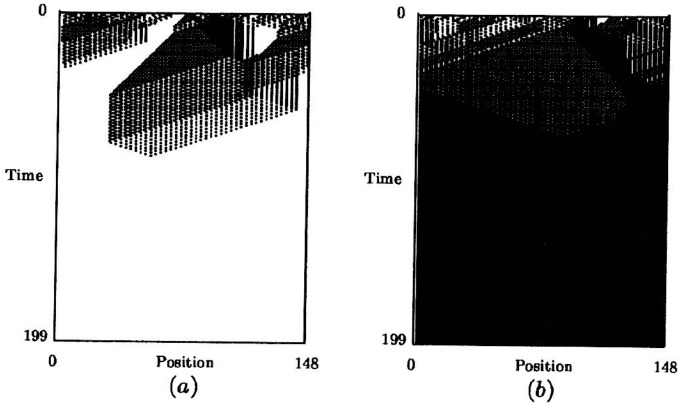

Fig. 1. These are examples of majority problem classification by the rule found by David, Forrest and Koza. [5]. Both are correct classifications (a) with 74 ones in the initial state, (b) with 75.

every cell in the CA is zero and the 'all one' state being a the state in which every cell is one.

Evaluating a transition rule for this problem is done by iterating M randomly generated initial states and calculating the relative number of correct classification. The fitness of a transition rule is denoted with $F_{N,M}$ where N is the width of the CA. The fitness can be calculated with different distributions over the number of ones in the initial state, but the default is a binomial distribution (denoted with $F^B_{N,M}$) where every cell in the CA has a 50% chance of being initiated with a one for every initial state.

The first intuitive rule to come up with is the 'majority rule'. This being the rule where the output value is 1, if the number of ones in the neighborhood is more than the number of zeros, and a zero otherwise. Surprising as it may seem this does not at all solve the problem. The majority rule gets stuck on the problem that on the boundary thick line in the time plot the cell can't "agree" on the global answer. The cell just left of such a thick line is zero and because all other cells left of it in the neighborhood are also zero, it "decides" to stay that way. Yet its neighboring cell to the right is one and sees only ones on its right and therefore decides to stay one. This way the information fails to propagate through the CA and classification fails.

Researchers in the field of cellular automata have published many different rules to solve this problem, one such rule is the GKL rule after Gacs, Kurdyumov and Levin [8]. This rule is pretty good at classifying the majority problem and does it for 81.6% of the test cases with a width of 149 cells. For 17 years this was the best rule and then L. Davis found a better one in 1995 which did 81.8%. In the same year R. Das found a rule that did 82.178%. Then in 1996 David, Forrest and Koza found a rule by cleverly using genetic programming that was able to classify 82.326% correctly [5].

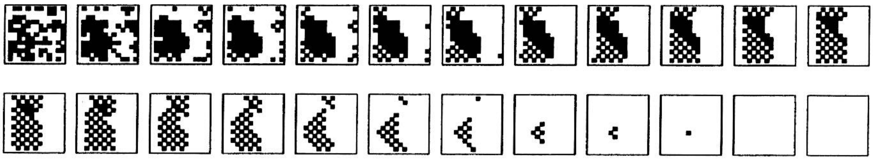

Fig. 2. This figure shows a correct classification of the Majority Problem by a two dimensional CA with both width and height equal to 13 and $\lambda = 84/169$. The transition rule was one of the best tested in the experiment and scored $F_{169,10^3} = 0.715$.

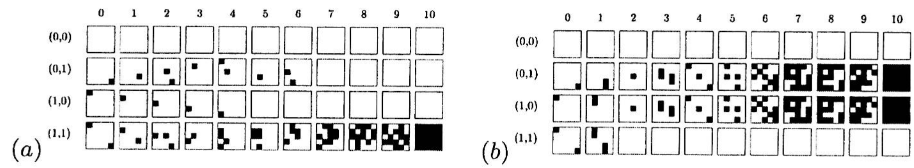

Fig. 3. This figure shows two iteration runs of transition rules for two dimensional CA that were evolved using a GA. (*a*) shows how a CA can behave like an AND-port and (*b*) shows how it can behave like an XOR-port.

Although these rules are very impressive it is believed that there is no definite solution for the problem as long as the neighborhood is smaller than the size of the CA. It is already a big accomplishment for a CA to get 70 percent of all random initial states correct, for this shows there is some kind of communication going on; some kind of emerging behavior.

4 Inverse Design

Nature has some remarkable examples of local rules that exhibit global behavior. Ant colonies are a good example. An individual ant does not seem to be very intelligent and does not seem to know what is doing and why, but a colony of ants seem very organized and purposeful. This did not happen over night, but evolved over millions of generations.

The global behavior of a CA can also be evolved in a similar way by evolving the local behavior. This "inverse design" of the behavior is done by using a genetic algorithm to evolve the transition rule that described the behavior. The fitness function of the genetic algorithm then defines the desired behavior.

M. Mitchell, J. P. Crutchfield and P. T. Hraber have shown [14,15] that using a simple GA to evolve transition rules for the majority problem (explained in Section 3) can already give surprisingly good results. About half of the rules that were found performed better than the most trivial rule and about 7 rules out of 300 rules were found that seemed to use some primitive form of communication that worked for more than 70% of the classifications. This is not better than