

Stefania Bandini
Sara Manzoni
Hiroshi Umeo
Giuseppe Vizzari (Eds.)

LNCS 6350

Cellular Automata

9th International Conference on Cellular Automata
for Research and Industry, ACRI 2010
Ascoli Piceno, Italy, September 2010, 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

Stefania Bandini Sara Manzoni
Hiroshi Umeo Giuseppe Vizzari (Eds.)

Cellular Automata

9th International Conference on Cellular Automata
for Research and Industry, ACRI 2010
Ascoli Piceno, Italy, September 21-24, 2010
Proceedings

Volume Editors

Stefania Bandini

Complex Systems and Artificial Intelligence Research Center
University of Milano-Bicocca

Viale Sarca 336/14, Milano, Italy

E-mail: stefania.bandini@disco.unimib.it

Sara Manzoni

Complex Systems and Artificial Intelligence Research Center
University of Milano-Bicocca

Viale Sarca 336/14, Milano, Italy

E-mail: sara.manzoni@disco.unimib.it

Hiroshi Umeo

Faculty of Information Science and Technology

University of Osaka Electro-Communication

572-8530 Neyagawa, Osaka, Japan

E-mail: umeo@cyt.osaka.ac.jp

Giuseppe Vizzari

Complex Systems and Artificial Intelligence Research Center
University of Milano-Bicocca

Viale Sarca 336/14, Milano, Italy

E-mail: giuseppe.vizzari@disco.unimib.it

Library of Congress Control Number: 2010934512

CR Subject Classification (1998): F.1.1, F.1, F.2.2, I.6, C.2

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

ISSN 0302-9743

ISBN-10 3-642-15978-8 Springer Berlin Heidelberg New York

ISBN-13 978-3-642-15978-7 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© Springer-Verlag Berlin Heidelberg 2010

Printed in Germany

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

Printed on acid-free paper 06/3180

Foreword

This volume collects the papers selected for presentation at the 9th International Conference on Cellular Automata for Research and Industry (ACRI 2010), held in Ascoli Piceno (Italy), September 21-24, 2010.

ACRI conferences have been offering since 1994 a biennial scientific meeting to both scientists and innovation managers in academia and industry to express and discuss their viewpoints on current and future trends, challenges, and state-of-the-art solutions to various problems in the fields of arts, biology, chemistry, communication, ecology, economy, engineering, networks, physics, social science, and traffic control. ACRI 2010 was organized by the Complex Systems and Artificial Intelligence (CSAI) research center of the University of Milano-Bicocca as a forum for the presentation and discussion of specialized results as well as general contributions to the growth of the cellular automata approach and its application. Cellular automata represent a very powerful approach to the study of spatio-temporal systems where complex phenomena are built up out of many simple local interactions. The ACRI conference series was first organized in Italy (ACRI 1994 in Rende, ACRI 1996 in Milan, and ACRI 1998 in Trieste), and after having moved to other European and international settings, this year came back to Italy: ACRI 2000 in Karlsruhe (Germany), ACRI 2002 in Geneva (Switzerland), ACRI 2004 in Amsterdam (The Netherlands), ACRI 2006 in Perpignan (France), and ACRI 2008 in Yokohama (Japan).

In order to give a perspective in which both theoretical and applicational aspects of cellular automata contribute to the growth of the area, this book mirrors the structure of the conference, grouping the 74 papers into two main parts. The first part collects papers presented as part of the main conference and organized according to six main topics: (1) theoretical results on cellular automata, (2) modeling and simulation with cellular automata, (3) CA dynamics, control and synchronization, (4) codes and cryptography with cellular automata, (5) cellular automata and networks, and (6) CA-based hardware. The second part of the volume is dedicated to contributions presented during the ACRI 2010 workshops on theoretical advances, specifically Asynchronous Cellular Automata (chairs: Alberto Dennunzio, Enrico Formenti, and Marco Tomassini), and challenging application contexts for cellular automata: Crowds and CA (3rd edition, chairs: Sara Manzoni and Shin Morishita), Traffic and CA (chairs: Katsuhiko Nishinari and Andreas Schadschneider), and the International Workshop of Natural Computing (chairs: Ferdinand Peper and Hiroshi Umeo).

Many people contributed to the success of ACRI 2010 and to the creation of this volume, from the initial idea to its implementation. Our first acknowledgement is to all the scientists that submitted their works, and to all Program

Committee members and reviewers for their precious collaboration. A special thanks for their hospitality to the University of Camerino (in particular to Emanuela Merelli and Flavio Corradini), Centro Studi Piceno, the municipality of Ascoli Piceno, and Fondazione Casse di Risparmio della Provincia di Ascoli Piceno, and for its generous contribution to the realization of this volume to the University of Milano-Bicocca.

A special acknowledgement also to all the people involved in the organization of ACRI 2010 (in particular to Paola Lembo, Giorgia Malvolta, Lorenza Manenti, and Luca Manzoni) whose work was fundamental for the actual success of the event.

Finally, we would like to thank the Department of Computer Science, Systems and Communication of the University of Milano–Bicocca, and those that financially supported the congress: illycaffè and Fondazione Casse di Risparmio della Provincia di Ascoli Piceno.

July 2010

Stefania Bandini
Sara Manzoni
Hiroshi Umeo
Giuseppe Vizzari

Organization

ACRI 2010 was organized by the Complex Systems and Artificial Intelligence research center (CSAI) and Artificial Intelligence Lab (L.INT.AR.) of the Department of Computer Science, Systems and Communication (DISCo) of the University of Milano–Bicocca, and hosted by the University of Camerino.

Organizing Committee

Sara Manzoni (Chair), Paola Lembo, Giorgia Malvolta, Lorenza Manenti, Luca Manzoni, Emanuela Merelli, Giuseppe Vizzari.

Congress Chairs

Stefania Bandini	University of Milano-Bicocca
Hiroshi Umeo	University of Osaka

Steering Committee

Stefania Bandini	University of Milano-Bicocca, Italy
Bastien Chopard	University of Geneva, Switzerland
Giancarlo Mauri	University of Milano-Bicocca, Italy
Hiroshi Umeo	University of Osaka, Japan
Thomas Worsch	University of Karlsruhe, Germany

Program Committee

Susumu Adachi (Japan)	Nazim Fats (France)
Andrew Adamatzky (UK)	Alfons Hoekstra (The Netherlands)
Franco Bagnoli (Italy)	Teijiro Isokawa (Japan)
Stefania Bandini (Italy)	Francisco Jimnez (Spain)
Olga Bandman (Russia)	Toshihiko Komatsuzaki (Japan)
Belgacem Ben Youssef (Canada)	Anna T. Lawniczak (Canada)
Debashish Chowdhury (India)	Jia Lee (Japan)
Bastien Chopard (Switzerland)	Pradipta Maji (India)
Alberto Dennunzio (Italy)	Danuta Makowiec (Poland)
Andreas Deutsch (Germany)	Sara Manzoni (Italy)
Salvatore Di Gregorio (Italy)	Nobuyuki Matsui (Japan)
Michel Droz (Switzerland)	Giancarlo Mauri (Italy)
Samira El Yacoubi (France)	Michael Meyer-Hermann (Germany)

Angelo Mingarelli (Canada)	Domenico Talia (Italy)
Shin Morishita (Japan)	Gianluca Tempesti (Switzerland)
Katsuhiko Nishinari (Japan)	Marco Tomassini (Switzerland)
Hidenosuke Nishio (Japan)	Leen Torenvliet (The Netherlands)
Ferdinand Peper (Japan)	Hiroshi Umeo (Japan)
Roberto Serra (Italy)	Giuseppe Vizzari (Italy)
Georgios Sirakoulis (Greece)	Burton Voorhees (Canada)
Furio Suggi Liverani (Italy)	Thomas Worsch (Germany)
Peter Sloat (The Netherlands)	
G. Keith Still (UK)	

Workshop Chairs

Asynchronous CA	Alberto Dennunzio (Università di Milano-Bicocca, Italy) Enrico Formenti (Université de Nice - Sophia Antipolis, France) Marco Tomassini (Université de Lausanne, Switzerland)
Crowds and CA	Sara Manzoni (Università di Milano-Bicocca, Italy) Shin Morishita (Yokohama National University, Japan)
Traffic and CA	Katsuhiko Nishinari (University of Tokyo, Japan) Andreas Schadschneider (Universität zu Köln, Germany)
Natural Computing	Ferdinand Peper (National Institute of Information and Communications Technology, Japan) Hiroshi Umeo (University of Osaka Electro-Communication, Japan) Yasuhiro Suzuki (Nagoya University, Japan)

Table of Contents

Theoretical Results on Cellular Automata

Information Transfer among Coupled Random Boolean Networks	1
<i>Chiara Damiani, Stuart A. Kauffman, Roberto Serra, Marco Villani, and Annamaria Colacci</i>	
Open Environment for 2d Lattice-Grain CA	12
<i>Guillaume Cottenceau and Dominique Désérable</i>	
All-to-All Communication with CA Agents by Active Coloring and Acknowledging	24
<i>Patrick Ediger and Rolf Hoffmann</i>	
The Sandpile Model: Parallelization of Efficient Algorithms for Systems with Shared Memory	35
<i>Sebastian Frehmel</i>	
Theory and Application of Equal Length Cycle Cellular Automata (ELCCA) for Enzyme Classification	46
<i>Soumyabrata Ghosh, Tirthankar Bachhar, Nirmalya S. Maiti, Indrajit Mitra, and P. Pal Chaudhuri</i>	
Cellular Automata Model for Size Segregation of Particles	58
<i>Toshihiko Komatsuzaki and Yoshio Iwata</i>	
Convex Hulls on Cellular Automata	69
<i>Luidnel Maignan and Frédéric Gruau</i>	
Square Kufic Pattern Formation by Asynchronous Cellular Automata . . .	79
<i>Seyyed Amir Hadi Minoofam and Azam Bastanfard</i>	

Modeling and Simulation with Cellular Automata

Development and Calibration of a Preliminary Cellular Automata Model for Snow Avalanches	83
<i>Maria Vittoria Avolio, Alessia Errera, Valeria Lupiano, Paolo Mazzanti, and Salvatore Di Gregorio</i>	
Tracking Uncertainty in a Spatially Explicit Susceptible-Infected Epidemic Model	95
<i>Jan M. Baetens and Bernard De Baets</i>	
A Proximal Space Approach for Embedding Urban Geography into CA Models	106
<i>Ivan Blečić, Arnaldo Cecchini, and Giuseppe A. Trunfio</i>	

Bone Remodelling: A Complex Automata-Based Model Running in BIOSHAPE	116
<i>Diletta Cacciagrano, Flavio Corradini, and Emanuela Merelli</i>	
CANv2: A Hybrid CA Model by Micro and Macro-dynamics Examples	128
<i>Claudia R. Calidonna, Adele Naddeo, Giuseppe A. Trunfio, and Salvatore Di Gregorio</i>	
Simulation of Traffic Flow at a Signalised Intersection	138
<i>Somayyeh Belbasi and M. Ebrahim Foulaadvand</i>	
A Novel Method for Simulating Cancer Growth	142
<i>Mehrdad Ghaemi, Omid Naderi, and Zahra Zabihinpour</i>	
Towards Cellular Automata Football Models with Mentality Accounting	149
<i>Alexander Makarenko, Dmitry Krushinski, Anton Musienko, and Boris Goldengorin</i>	
The Complexity of Three-Dimensional Critical Avalanches	153
<i>Carolina Mejía and J. Andrés Montoya</i>	
Using Cellular Automata on a Graph to Model the Exchanges of Cash and Goods	163
<i>Ranaivo Mahaleo Razakanirina and Bastien Chopard</i>	
Montebello: A Metapopulation Based Model of Carcinogenesis	173
<i>David Tuck, Willard Miranker, and Jose Costa</i>	
 CA Dynamics, Control and Synchronization	
Towards Generalized Measures Grasping CA Dynamics	177
<i>Jan M. Baetens and Bernard De Baets</i>	
Synchronization and Control of Cellular Automata	188
<i>Franco Bagnoli, Samira El Yacoubi, and Raúl Rechtman</i>	
Discovery by Genetic Algorithm of Cellular Automata Rules for Pattern Reconstruction Task	198
<i>Anna Piwonska and Franciszek Seredynski</i>	
Addition of Recurrent Configurations in Chip Firing Games: Finding Minimal Recurrent Configurations with Markov Chains	209
<i>Matthias Schulz</i>	
A Seven-State Time-Optimum Square Synchronizer	219
<i>Hiroshi Umeo and Keisuke Kubo</i>	

Codes and Cryptography with Cellular Automata

Null Boundary 90/150 Cellular Automata for Multi-byte Error Correcting Code	231
<i>Jaydeb Bhaumik, Dipanwita Roy Chowdhury, and Indrajit Chakrabarti</i>	
Generating Cryptographically Suitable Non-linear Maximum Length Cellular Automata	241
<i>Sourav Das and Dipanwita Roy Chowdhury</i>	
Chaotic Cellular Automata with Cryptographic Application	251
<i>Amparo Fúster-Sabater and Pino Caballero-Gil</i>	
d-Monomial Tests of Nonlinear Cellular Automata for Cryptographic Design	261
<i>Sandip Karmakar, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury</i>	
Programmable Cellular Automata (PCA) Based Advanced Encryption Standard (AES) Hardware Architecture	271
<i>Nirmalya S. Maiti, Soumyabrata Ghosh, Biplab K. Shikdar, and P. Pal Chaudhuri</i>	
Exhaustive Evaluation of Radius 2 Toggle Rules for a Variable-Length Cryptographic Cellular Automata-Based Model	275
<i>Gina M.B. Oliveira, Luiz G.A. Martins, Leonardo S. Alt, and Giordano B. Ferreira</i>	

Cellular Automata and Networks

Network Decontamination with Temporal Immunity by Cellular Automata	287
<i>Yassine Daadaa, Paola Flocchini, and Nejb Zaguia</i>	
Characterization of CA Rules for SACA Targeting Detection of Faulty Nodes in WSN	300
<i>Sukanta Das, Nazma N. Naskar, Sukanya Mukherjee, Mamata Dalui, and Biplab K. Sikdar</i>	
Cellular Automata Applied in Remote Sensing to Implement Contextual Pseudo-fuzzy Classification	312
<i>Moisés Espínola, Rosa Ayala, Saturnino Leguizamón, Luis Iribarne, and Massimo Menenti</i>	
Impact of Coupling of Distributed Denial of Service Attack with Routing on Throughput of Packet Switching Network	322
<i>Anna T. Lawniczak, Hao Wu, and Bruno Di Stefano</i>	

CA-Based Hardware

A Cellular Automata-Based Modular Lighting System 334
Stefania Bandini, Andrea Bonomi, Giuseppe Vizzari, and Vito Acconci

Modeling and Programming Asynchronous Automata Networks: The MOCA Approach 345
Stefania Bandini, Andrea Bonomi, and Giuseppe Vizzari

Efficient Circuit Construction in Brownian Cellular Automata Based on a New Building-Block for Delay-Insensitive Circuits 356
Jia Lee and Ferdinand Peper

A Cellular Automaton Controlled Shading for a Building Facade 365
Machi Zawidzki

FPGA Design of a Cellular Automaton Model for Railway Traffic Flow with GPS Module 373
Anastasio Tsiftsis, Georgios Ch. Sirakoulis, and John Lygouras

ACA - Int. Workshop on Asynchronous CA

What Do We Mean by Asynchronous CA? A Reflection on Types and Effects of Asynchronicity 385
Stefania Bandini, Andrea Bonomi, and Giuseppe Vizzari

Parallel Composition of Asynchronous Cellular Automata Simulating Reaction Diffusion Processes 395
Olga Bandman

Comparative Study of Parallel Algorithms for Asynchronous Cellular Automata Simulation on Different Computer Architectures 399
Konstantin Kalgin

Coxeter Groups and Asynchronous Cellular Automata 409
Matthew Macauley and Henning S. Mortveit

Some Formal Properties of Asynchronous Cellular Automata 419
Luca Manzoni

A Study on the Automatic Generation of Asynchronous Cellular Automata Rules by Means of Genetic Algorithms 429
Andrea Valsecchi, Leonardo Vanneschi, and Giancarlo Mauri

C&CA - Int. Workshop on Crowds and CA

Towards Patterns of Comfort: A Multilayered Model Based on Situated Multi-agent Systems	439
<i>Paola Lembo, Lorenza Manenti, and Sara Manzoni</i>	
A Pedestrian Movement Model That Takes into Account the Capacity Drop Phenomenon in the Motion of Crowd	446
<i>Elvezia M. Cepolina and Alessandro Farina</i>	
A Cellular Automaton Model for Crowd Evacuation and Its Auto-Defined Obstacle Avoidance Attribute	455
<i>Ioakeim G. Georgoudas, Georgios Koltsidas, Georgios Ch. Sirakoulis, and Ioannis Th. Andreadis</i>	
A Learning Algorithm for the Simulation of Pedestrian Flow by Cellular Automata	465
<i>Hideaki Ishii and Shin Morishita</i>	
On Influencing of a Space Geometry on Dynamics of Some CA Pedestrian Movement Model	474
<i>Ekaterina Kirik, Tat'yana Yurgel'yan, and Dmitriy Krouglov</i>	
The Dynamic Distance Potential Field in a Situation with Asymmetric Bottleneck Capacities	480
<i>Tobias Kretz</i>	
Solving the Direction Field for Discrete Agent Motion	489
<i>Michael Schultz, Tobias Kretz, and Hartmut Fricke</i>	
Phase Coexistence in Congested States of Pedestrian Dynamics	496
<i>Armin Seyfried, Andrea Portz, and Andreas Schadschneider</i>	
Stochastic Transition Model for Discrete Agent Movements	506
<i>Michael Schultz and Hartmut Fricke</i>	
Analysis of Obstacle Density Effect on Pedestrian Congestion Based on a One-Dimensional Cellular Automata	513
<i>Kenichiro Shimura, Yuki Tanaka, and Katsuhiro Nishinari</i>	
Excluded Volume Effect in a Pedestrian Queue	523
<i>Daichi Yanagisawa, Yuki Tanaka, Rui Jiang, Akiyasu Tomoeda, Kazumichi Ohtsuka, Yushi Suma, and Katsuhiro Nishinari</i>	

T&CA - Int. Workshop on Traffic and CA

Simulation on Vehicle Emission by the Brake-Light Cellular Automata Model	532
<i>Liyun Dong, Peng Zhang, and Shiqiang Dai</i>	

Bidirectional Traffic on Microtubules	542
<i>Maximilian Ebbinghaus, Cécile Appert-Rolland, and Ludger Santen</i>	
Cellular Automata for a Traffic Roundabout	552
<i>Ding-wei Huang</i>	
Cellular Automata for a Cyclic Bus	557
<i>Ding-wei Huang and Wei-neng Huang</i>	
Dynamics of a Tagged Particle in the Asymmetric Exclusion Process with Particlewise Disorder	561
<i>Takashi Imamura</i>	
Chase and Escape in Groups	570
<i>Atsushi Kamimura, Shigenori Matsumoto, Nobuyasu Ito, and Toru Ohira</i>	
A Velocity-Clearance Relation in the Rule-184 Cellular Automaton as a Model of Traffic Flow	580
<i>Masahiro Kanai</i>	
CA and MAS – With the NaSch as Example	589
<i>Tobias Kretz</i>	
Productivity Enhancement through Lot Size Optimization	593
<i>Takumi Minemura, Katsuhiko Nishinari, and Andreas Schadschneider</i>	
Multilane Single GCA-w Based Expressway Traffic Model	600
<i>Anna T. Lawniczak and Bruno Di Stefano</i>	
Properties of Cellular Automaton Model for On-Ramp System	613
<i>Xin-Gang Li, Zi-You Gao, and Bin Jia</i>	
Inversion of Flux between Zipper and Non-Zipper Merging in Highway Traffic	619
<i>Ryosuke Nishi, Hiroshi Miki, Akiyasu Tomoeda, Daichi Yanagisawa, and Katsuhiko Nishinari</i>	
Clustering and Transport Efficiency in Public Conveyance System	625
<i>A. Tomoeda, R. Nishi, and K. Nishinari</i>	
Clusters in the Helbing’s Improved Model	633
<i>Rosa María Velasco and Patricia Saavedra</i>	
Phase Transitions in Cellular Automata for Cargo Transport and Kinetically Constrained Traffic	637
<i>Marko Woelki</i>	

A New Computational Methodology Using Infinite and Infinitesimal Numbers	646
<i>Yaroslav D. Sergeyev</i>	
IWNC - Int. Workshop on Natural Computing	
Molecular Implementations of Cellular Automata	650
<i>Satyajit Sahu, Hiroshi Oono, Subrata Ghosh, Anirban Bandyopadhyay, Daisuke Fujita, Ferdinand Peper, Teijiro Isokawa, and Ranjit Pati</i>	
Achieving Universal Computations on One-Dimensional Cellular Automata	660
<i>Jean-Baptiste Yunès</i>	
Author Index	671

Information Transfer among Coupled Random Boolean Networks

Chiara Damiani¹, Stuart A. Kauffman², Roberto Serra¹,
Marco Villani¹, and Annamaria Colacci³

¹ Department of Social, Cognitive and Quantitative Sciences, Modena and Reggio Emilia University, via Allegri 9; I-42100 Reggio Emilia
{chiara.damiani, rserra, mvillani}@unimore.it

² Departments of Biochemistry and Mathematics
University of Vermont, Burlington, VT 05405
stuart.kauffman@uvm.edu

³ Excellence Environmental Carcinogenesis, Environmental Protection and Health Prevention Agency, Emilia-Romagna, viale Filopanti 22, Bologna, Italia
acolacci@arpa.emr.it

Abstract. Information processing and information flow occur at many levels in the course of an organism's development and throughout its lifespan. Biological networks inside cells transmit information from their inputs (e.g. the concentrations of proteins or other signaling molecules) to their outputs (e.g. the expression levels of various genes). Moreover, cells do not exist in isolation, but they constantly interact with one another. We study the information flow in a model of interacting genetic networks, which are represented as Boolean graphs. It is observed that the information transfer among the networks is not linearly dependent on the amount of nodes that are able to influence the state of genes in surrounding cells.

Keywords: Cellular Automata, Random Boolean Networks, Mutual Information, Transfer Entropy, Cellular Communication.

1 Introduction

An active area of research in the field of complex systems is the study of their information storing and processing capabilities. The amount of information that a system is able to elaborate and store plays indeed a crucial role in networks of many interacting units such as gene regulatory networks, social networks or economic networks.

Random Boolean Networks (RBNs) have been employed to study under what conditions the information processing ability of complex systems is optimized [1,2,3,4,5,6].

Random Boolean networks were initially introduced as a simplified model of genetic regulatory networks [7]. The most interesting feature of RBNs is that their dynamics can be classified as ordered, disordered, or critical [8,9,10]. The

wide availability of gene expression data has allowed interesting comparisons between the behavior of these models and that of real cells, such as the distribution of perturbations induced by gene knock-outs [11] and the time course of synchronized leukemia cells [12]. Despite their abstract properties, random Boolean networks have proven to be able to describe significant quantitative features of biological systems and have given indications in favour of the hypothesis that real cells might be operating in the critical region. This seems surprising, especially in view of the discrete approximation. Although models based on ordinary differential equations may provide a better description of gene regulation phenomena, for computational reasons, the use of simplified Boolean models has the unique advantage that allows to deal with larger networks. Furthermore, it has been observed that - e.g. while describing cell differentiation [13] - Boolean model share some features with important continuum models of the same phenomena.

Several authors in the last few years have shown that critical RBNs optimize the information flow and coordination of behavior diversity among its nodes [1,2,3,4,5,14]. We aim here at investigating the conditions for such optimization, but among RBNs that are put into communication with one another.

We have previously introduced Multi Random Boolean Networks [15,16,17,18] as a model for the interaction of cells based on a Cellular Automaton of locally interacting RBNs. It may also serve as an interesting general example of interconnected homologous systems that are able to show emergent dynamical behaviors. The interaction among cells is obtained by letting the activation of some nodes be affected by the activation of some other nodes in adjacent cells. We have conceived two alternative updating schemes, which we refer to as molecule-sharing and molecule-signaling. The first mechanism roughly imitates the diffusion of gene products to surrounding cells, the second one is inspired to the most common form of cell communication, according to which a signaling cell secretes a molecule and the target cell has a specific receptor that binds to the signaling molecule, thus triggering a change in the expression level of downstream genes. So far, the two methods have not shown relevant differences with respect to the main properties of some measures that we considered in past studies [18]. Since also the information measures proposed in this work seem to be robust to the choice of the mechanism, as observed in preliminary results, we will focus our analysis on one of the two updating rules, specifically on molecule-sharing. The mechanism is described in Section 2.

The intensity of the coupling among the networks can be modified by varying the fraction of nodes of each RBN that is able to communicate with neighboring RBNs. In this work we study how the information transfer among RBNs that are built with critical parameters is affected by such variable.

In information theory, a key measure of the information in a random variable is the entropy, which indicates how easily message data can be compressed. Whereas the most important measure of the amount of information in common between two random variables is the mutual information.

Within RBNs, the entropy of a node is related to its activity. In the ordered regime, almost all nodes are frozen, in the chaotic regime, on the other hand,

the number of fluctuating nodes is a finite fraction of N [19]. It therefore appears important to understand how the frozen component of critical RBNs varies when they are put into communication with one another. We introduce also a measure of the velocity with which a node fluctuates among different states: the variability of a node.

In RBNs the average of the mutual information over all pairs has been proposed as a global measure of how well the system can coordinate its internal dynamics. We extend this measure to quantify the amount of information exchanged between nodes belonging to different interacting RBNs. We also study a simpler measure of correlation between them and a recently proposed alternative measure of information transfer, i.e. the transfer entropy. All the above mentioned measures are defined in Section 3.

We numerically analyze how such measures are affected by the coupling strength. The results reported in Section 4 show that the relationship between the intensity of the coupling and the amount of information either contained in the nodes of each cell or transferred among the cells is not trivial, on the contrary it shows a minimum point for a moderate level of the coupling strength.

2 MRBN Model

A MRBN is a two dimensional Cellular Automaton where a complete RBN is placed in each site. The new state of each RBN at time $t+1$ is determined by the current state of its n nodes and the state of some nodes belonging to its four orthogonal neighbors, according to a fixed rule. The updating rule is identical for each cell, does not change over time, and is simultaneously applied to the whole grid. Periodic boundaries conditions are used.

Every cell x has a set G of n nodes or genes $G = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$. Each gene i is represented as a Boolean logic processing unit that receives inputs from k_i other genes $q_{1(i)}, q_{2(i)}, \dots, q_{k_i(i)}$. The inputs to each gene are chosen at random among all the other genes of the network. The Boolean function $f_i(t) \equiv f_i(\sigma_{q_{1(i)}}(t), \sigma_{q_{2(i)}}(t), \dots, \sigma_{q_{k_i(i)}}(t))$ is associated to gene i by randomly selection from a distribution of all the possible Boolean rules with k_i inputs. Node i have the same inputs and same Boolean function in all the m cells of the automaton. In this study we set $k_i = 2, \forall i = 1, 2, \dots, n$ and a uniform distribution of the Boolean functions. These parameters delimitate the boundary between ordered and chaotic regime in not coupled classical RBNs [8,9,10].

Only a subset $S \subset G$ of the n genes is not exclusively controlled by its k_i inputs according to its Boolean function, but by nodes in surrounding cells as well. We refer to these s nodes as *communicating nodes*. Note that the set of communicating nodes is identical in every cell. If i is a communicating node ($i \in S$) and the output of its Boolean function in cell x is 0, but in a given fraction of neighboring cells it is 1, gene i will take value 1 in cell x as well.

Formally: let W_x be the set of four orthogonal neighbors of cell x . Let $\theta_{i,x}(t)$ be the fraction of neighboring cells of x in which the function of gene i has value 1 at time t and let it be called the ‘‘concentration’’ of gene i in cell x at time t .

$$\theta_{i,x}(t) = \frac{1}{4} \sum_{w \in W_x} f_{i,w}(t) \quad (1)$$

The value of gene i at time $t+1$ depends also on the value $\theta_{i,x}(t)$ according to a *concentration function* $g(\theta_{i,x}(t))$:

$$g(\theta_{i,x}(t)) = \begin{cases} 0 & \text{if } \theta_{i,x}(t) \leq \tau \\ 1 & \text{otherwise} \end{cases}, 0 \leq \tau \leq 1 \quad (2)$$

where τ is a fixed threshold value, and it is identical for every communicating node.

The dynamics of each graph is described by the following equation:

$$\sigma_{i,x}(t+1) = \begin{cases} f_i(t) & \text{if } i \notin S \\ f_i(t) & \text{if } i \in S \text{ and } g(\theta_{i,x}(t)) = 0 \\ 1 & \text{if } i \in S \text{ and } g(\theta_{i,x}(t)) = 1 \end{cases} \quad (3)$$

In this study, we limit our analysis to MRBNs with $\tau = 0$, assuming that the expression of a communicating gene in at least one of the neighbors is sufficient condition for the interaction to occur.

We define the parameter χ - *coupling strength* - as the fraction of communicating nodes over the n total number of nodes: $\chi = s/n$.

Since the MRBN dynamics is deterministic, after a transient it will evolve to an attractor. In the attractor, every cell will always repeat the same configuration of states, which we refer to as sub-attractor. We let the networks evolve from different initial conditions; as a consequence the m cells can be found in a different sub-attractors, where a is a number between 1 and m . The magnitude of a is affected by the communication among the cells. We have observed that this relationship is not monotonous [18]; on the contrary, the number of different sub-attractors in MRBNs made of 5×5 cells is maximized for a level of the coupling strength around 0.10. For higher levels of the coupling strength the cells tend instead to become homogenous.

3 Information Processing and Transfer Measures

The average *activity* of a node characterizes its typical dynamical behavior. The activity of node i in cell x is defined as [14]:

$$A_{i,x} = 1 - \left| \frac{1}{T} \sum_{t=1}^T (2\sigma_{i,x}(t) - 1) \right| \quad (4)$$

where $\sigma_{i,x}$ is the value of gene i in cell x and T is the length of the time series over which the activity is measured. The so-called frozen nodes, which never change their state over time, have an activity $A_{i,x} = 0$, whereas the nodes that occasionally change their state have an activity $0 < A_{i,x} \leq 1$.

The *frozen component* F [6], defined as the fraction of nodes whose average activity is 0, is a decisive quantity for the information flow among cells. If all the

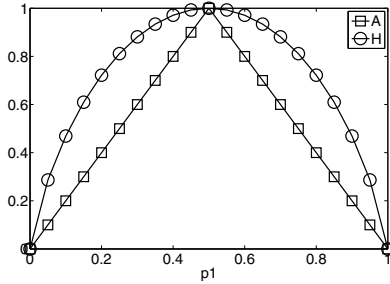


Fig. 1. Entropy and Activity as a function of p_1

nodes are frozen ($F = 1$), the network is basically irresponsive to signals from the environment and hence can neither process information nor adapt.

The average activity of a node is related to the information content of its message as expressed by the Shannon entropy in information theory [20]. The Shannon entropy is a measure of the uncertainty associated with a random variable. When the variable is Boolean it is defined as:

$$H_{i,x} = -p_1(\sigma_{i,x})\log_2 p_1(\sigma_{i,x}) - [1 - p_1(\sigma_{i,x})]\log_2[1 - p_1(\sigma_{i,x})], \quad (5)$$

where $p_1(\sigma_{i,x})$ is the probability of node $\sigma_{i,x}$ to have value 1. Similarly to the average activity, $H_{i,x}$ is maximized when $p_1 = 1/2$. Figure 1 compares the basic properties of the two measures. The activity of a node, as well as its entropy, provides information on the values a gene can take over time but does not take into account how often the gene oscillates between such values. For example, the case $A_{i,x} = 0.5$ indicates that gene i in cell x takes value 1 50% of the time and 0 the remaining time, but it does not specifies whether it oscillates between the two values every other time step or just once in the entire time series. We define therefore the *variability* of a node $V_{i,x}$ as:

$$V_{i,x} = \frac{1}{T} \sum_{t=1}^T |\sigma_{i,x}(t) - \sigma_{i,x}(t+1)| \quad (6)$$

$V_{i,x} = 0$ if the node is frozen and $V_{i,x} = 1$ if the node alternates its value at every time step.

Rohlf and Bornholdt [6] have introduced the average *correlation* in order to characterize the dynamical coordination of pairs of nodes. We extend its definition to nodes belonging to different cells of the MRBN. The average correlation between two nodes i and j respectively belonging to cell x and y is defined as:

$$C_{i,x;j,y} = \left| \frac{1}{T} \sum_{t=1}^T (2\sigma_{i,x}(t) - 1)(2\sigma_{j,y}(t) - 1) \right|, \quad x \neq y, \quad (7)$$

If the dynamics of the two nodes are correlated the two factors always have the same sign, if they are anti-correlated they always have opposite sign. In both

cases $C_{i,x;j,y}$ has value 1. If the relationship between the two genes varies in time $0 \leq C_{i,x;j,y} < 1$. Note that the magnitude of the average correlation is affected by the presence of frozen nodes.

The definition of entropy can be naturally extended to a pair of nodes in different cells. The joint entropy $H_{i,x;j,y}$ of node i in cell x and node j in cell y is defined as:

$$H_{i,x;j,y} = - \sum_{\alpha \in \{0,1\}} \sum_{\beta \in \{0,1\}} p_{\alpha,\beta}(\sigma_{i,x}, \sigma_{j,y}) \log_2 p_{\alpha,\beta}(\sigma_{i,x}, \sigma_{j,y}), x \neq y \quad (8)$$

We now introduce mutual information [20], which is a measure of the amount of information that a variable contains about another random variable. It is the reduction in the uncertainty of one random variable due to the knowledge of the other one. The mutual information between two variables is obtained as the relative entropy between their joint distribution and the product of their distributions. The relative entropy is a measure of the distance between two distributions, a measure of the inefficiency of assuming that the distribution is q when the true distribution is p . The relative entropy is always non negative and is zero if and only if $p = q$. We define the mutual information $I_{i,x;j,y}$ between node i in cell x and node j in cell y as:

$$I_{i,x;j,y} = \sum_{\alpha \in \{0,1\}} \sum_{\beta \in \{0,1\}} p_{\alpha,\beta}(\sigma_{i,x}, \sigma_{j,y}) \log_2 \frac{p_{\alpha,\beta}(\sigma_{i,x}, \sigma_{j,y})}{p_{\alpha}(\sigma_{i,x}) p_{\beta}(\sigma_{j,y})}, x \neq y \quad (9)$$

Mutual information has been used to infer biological relationships among genes from experimental data [21]. Nevertheless Mutual Information fails to distinguish information that is actually exchanged from shared information due to common history and input signals. Schreiber [22] has proposed an alternative information measure, called transfer entropy, that aims at overcoming these limitations while maintaining the desired properties of mutual information. The basic idea is to measure the divergence from independence between two genes i and j by observing their transition probabilities. In the absence of information flow from i to j , the state of i has no influence on the transition probabilities of system j . The transfer entropy measures the incorrectness of this assumption calculating the deviation from the generalized Markov property through the relative entropy. The transfer entropy from i to j is defined as [4]:

$$T_{i,x \rightarrow j,y} = \sum_{\gamma \in \{0,1\}} \sum_{\alpha \in \{0,1\}} \sum_{\beta \in \{0,1\}} p_{\gamma,\alpha,\beta}(\sigma_{i,x}(t+1), \sigma_{i,x}(t), \sigma_{j,y}(t)) \log_2 \frac{p_{\gamma,\alpha,\beta}(\sigma_{i,x}(t+1)|\sigma_{i,x}(t), \sigma_{j,y}(t))}{p_{\gamma,\alpha}(\sigma_{i,x}(t+1)|\sigma_{i,x}(t))}, x \neq y \quad (10)$$

¹ $p_{\gamma,\alpha,\beta}(\sigma_{i,x}(t+1), \sigma_{i,x}(t), \sigma_{j,y}(t))$ is the joint probability of $\sigma_{i,x}(t+1) = \gamma$, $\sigma_{i,x}(t) = \alpha$ and $\sigma_{j,y}(t) = \beta$. For a detailed explanation of the equation the reader is referred to [22].

4 Numerical Results

Since the measures described in the previous section are not analytically solvable yet, we numerically estimated them. The approximation accuracy increases with the length of the time series over which the quantities are computed. Since the attractor of the MRBN is cyclically repeated, the time series of the attractor is adequate for the estimation. Our simulation procedure is slightly different from the one of other similar studies on Random Boolean Networks [1,2,3,4,5,14]. Instead of collecting a long time series and discarding a partly arbitrary transient, we collect the precise time series of the attractor. MRBN that do not reach an attractor within the computational limit of 2500 time steps are discarded.

For each experiment, A_i, V_i, H_i are averaged over the total number of nodes of the MRBN (sometimes the set of nodes is restricted to communicating or non-communicating nodes), while the pairwise measures $C_{i,x;j,y}, T_{i,x;j,y}$ and $I_{i,x;j,y}$ are averaged over 10000 randomly chosen pairs. The measures typical of each coupling strength - $\langle A \rangle, \langle V \rangle, \langle H \rangle, \langle C \rangle, \langle I \rangle$ and $\langle T \rangle$ - are obtained by averaging over 100 randomly generated MRBNs made by 5x5 cells of 100 nodes each; and over 150 experiments from different randomly chosen initial states and communicating node sets per MRBN.

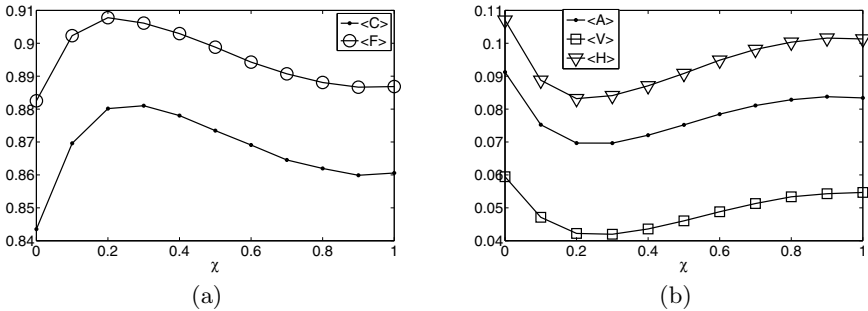


Fig. 2. (a) Average correlation $\langle C \rangle$ and frozen component $\langle F \rangle$ as a function of $\chi = 0, 0.1, 0.2, \dots, 1$ (b) Average activity $\langle A \rangle$, variability $\langle V \rangle$ and entropy $\langle H \rangle$ as a function of $\chi = 0, 0.1, 0.2, \dots, 1$

We analyzed the average correlation $\langle C \rangle$ and the average frozen component $\langle F \rangle$ as a function of the coupling strength (figure 2-a). It is remarkable how the average correlation is not linearly dependent on the coupling strength; on the contrary it shows a maximum point for a moderate level of the coupling strength, approximately around 0.20 and in any case between 0.10 and 0.30. As expected $\langle C \rangle$ is tightly related to the frozen component, it is maximized indeed when the number nodes that are not transmitting information is higher².

² The non coincidence of the two maximum points may be due to undersampling. Note that the two curves are the result of different experiments on the same set of MRBNs.

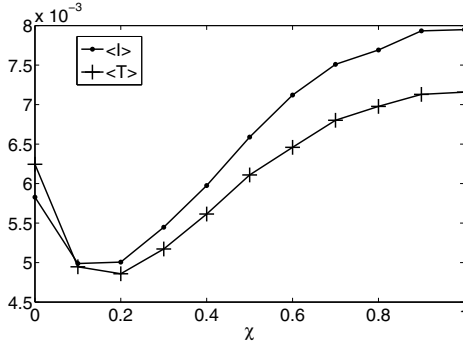


Fig. 3. Average pairwise Mutual Information $\langle I \rangle$ and average pairwise Transfer Entropy $\langle T \rangle$ as a function of $\chi = 0, 0.1, 0.2, \dots, 1$

Figure 2-b shows that also the average activity, variability and entropy are minimized for a modest level of χ . Counterintuitively, when such a fraction of nodes is communicating with neighboring cells, the capacity of nodes in general to process information is reduced and is lower than their capacity within isolated networks.

The reduced capacity to transmit and decode messages affects the ability of the system to propagate information and thus the coordination among the cells. The amount of mutual information contained in the time series of two elements gives a measure of how well they coordinate their activities. Figure 3 shows indeed that the average pairwise mutual information is minimized by a moderate coupling strength. Surprisingly, when the networks are isolated and communication is therefore impossible among them their mutual information is not null. Since the interacting networks are structurally identical, this phenomenon is plausibly due to the similarity of the processes. Regardless their history, nodes in differing networks are likely to show similar dynamics due to the fact that they are ruled by the same Boolean functions. Some functions may indeed give the same output for different configurations of the input values. This feature of the model implies that not even the transfer entropy is null when $\chi = 0$. It is shown in Figure 3 that this information transfer measure is minimized for a $0.10 < \chi < 0.30$ coupling as well.

We have seen how the typical ability of the nodes to propagate information is influenced by the coupling strength level. It is convenient to separately analyze the dynamical properties of communicating and non communicating nodes in order to discover if they are both responsible of the observed behaviors. One may indeed expect communicating nodes to have a lower activity due to the nature of their updating functions. The frozen component and average activity of communicating nodes and non communicating nodes as a function of the coupling strength, shown in Figure 4, reveal an intriguing phenomenon. As expected, communicating nodes tend to have a lower activity compared to non communicating ones. Nevertheless the two node types have a markedly different dependency

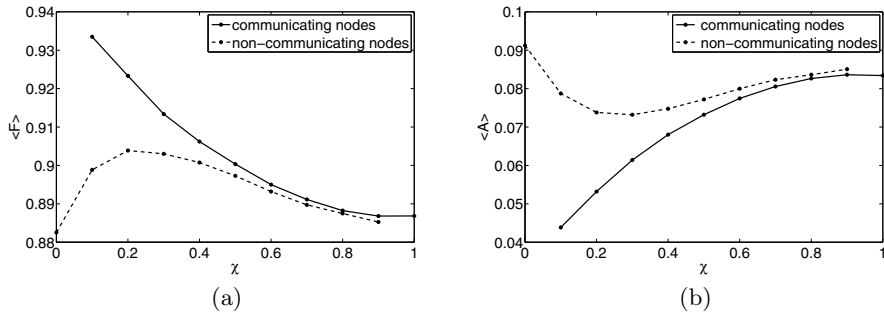


Fig. 4. Comparison between communicating and non-communicating nodes: frozen component $\langle F \rangle$ (a) and average activity $\langle A \rangle$ (b)

on the coupling strength. Consistently with the average behavior described before, non communicating nodes minimize their activity and maximize the frozen component when $0.10 < \chi < 0.30$. On the contrary, the activity and frozen component of communicating nodes respectively increase and decrease as a function of the coupling strength. It seems that the activity of the two types of node tend to converge when the coupling strength increases. Note that the proportions of the two node types vary with the coupling strength, producing the bowl-shaped average behaviors described in Figure 2. The dualism of the dynamics of communicating and non-communicating nodes is an emergent phenomenon that is difficult to predict starting from the updating rules of the models. The maximum of the activity of non communicating nodes at $0.10 < \chi < 0.30$ may be an outcome of the increased presence of feedbacks loops between communicating and not communicating nodes.

5 Conclusions

We have quantified, by means of numerical estimation of theoretical measures based on information theory, the amount of information that coupled RBNs are capable of transferring to each other.

We have examined the influence of the fraction of genes that are able to affect the state of surrounding cells (χ) on the information transfer among cells.

It has been observed that such relationship is not monotonous. On the one hand, the average correlation among the cells seems to be maximized for a moderate level of the coupling strength, on the other hand this maximum is an outcome of the reduced capability of the nodes to process information, their state being mainly frozen in time. As a consequence, the information transfer among the cells, measured by quantities such as mutual information and transfer entropy, is there minimized.

Although there are of course further information-theoretic measures which may be applied, the analysis has revealed that the behavior of the amount of information transferred among coupled cells is not intuitive and deserves further

investigations. The exact localization of its minimum may be the object of further research, along with its comparison with different concentration thresholds ($\tau > 0$); coupling strength being equal, a higher τ has indeed an impact on the actual realization of the communication among the cells.

Acknowledgments. This work has been partially supported by the Italian MIUR-FISR project nr. 2982/Ric (Mitica). Special thanks to the Institute for Bio-complexity and Informatics, at the University of Calgary (Canada), for warmly hosting the first author during six months.

References

1. Ribeiro, A.S., Este, R.A., Lloyd-Price, J., Kauffman, S.A.: Measuring information propagation and retention in boolean networks and its implications to a model of human organizations. In: Madureira, A.M. (ed.) Proceedings of the 6th WSEAS International Conference on Simulation, Modelling and Optimization, Lisbon, Portugal, September 22-24, pp. 391–398. WSEAS (2010)
2. Ribeiro, A.S., Kauffman, S.A., Lloyd-Price, J., Samuelsson, B., Socolar, J.E.S.: Mutual information in random boolean models of regulatory networks. *Physical Review E* 77, 011901 (2008)
3. Nykter, M., Price, N.D., Larjo, A., Aho, T., Kauffman, S.A., Yli-Harja, O., Shmulevich, I.: Critical networks exhibit maximal information diversity in structure-dynamics relationships. *Phys. Rev. Lett.* 100(5), 058702 (2008)
4. Krawitz, P., Shmulevich, I.: Basin entropy in boolean network ensembles. *Phys. Rev. Lett.* 98(15), 158701 (2007)
5. Luque, B., Ferrera, A.: Measuring mutual information in random boolean networks (1999)
6. Rohlf, T., Bornholdt, S.: Understanding Complex Systems. In: Self-Organized Criticality and Adaptation in Discrete Dynamical Networks, pp. 73–106. Springer, Heidelberg (2009)
7. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* 22(3), 437–467 (1969)
8. Kauffman, S.A.: At home in the universe. Oxford University Press, Oxford (1995)
9. Aldana, M.: Dynamics of boolean networks with scale free topology. *Physica D* (158), 45–66 (2003)
10. Derrida, B., Pomeau, Y.: Random networks of automata: a simple annealed approximation. *Europhys. Lett.* 1, 45–49 (1986)
11. Serra, R., Villani, M., Graudenzi, A., Kauffman, S.A.: On the distribution of small avalanches in random boolean networks. In: Ruusovori, P., et al. (eds.) Proceedings of the 4th TICSP workshop on computational systems biology, pp. 93–96. Juvenes print, Tampere (2006)
12. Shmulevich, I., Kauffman, S.A., Aldana, M.: Eukaryotic cells are dynamically ordered or critical but not chaotic. *PNAS* 102, 13439–13444 (2005)
13. Serra, R., Villani, M., Barbieri, A., Kauffman, S.A., Colacci, A.: On the dynamics of random boolean networks subject to noise: attractors, ergodic sets and cell types. *Journal of theoretical biology* (in press)
14. Andrecut, M., Foster, D., Carteret, H., Kauffman, S.A.: Maximal information transfer and behavior diversity in random threshold networks (2009)

15. Serra, R., Villani, M., Damiani, C., Graudenzi, A., Colacci, A., Kauffman, S.A.: Interacting random boolean networks. In: Proceedings of the European Conference on Complex Systems (ECCS 2007), Dresden, October 1-5 (2007)
16. Serra, R., Villani, M., Damiani, C., Graudenzi, A., Colacci, A.: The diffusion of perturbations in a model of coupled random boolean networks. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) ACRI 2008. LNCS, vol. 5191, pp. 315–322. Springer, Heidelberg (2009)
17. Damiani, C., Graudenzi, A., Villani, M.: How critical random boolean networks may be affected by the interaction with others. In: Serra, R., Villani, M., Poli, I. (eds.) Proceeding of WIVACE 2008, Artificial Life and evolutionary computation, pp. 271–283. World Scientific, Singapore (2008)
18. Damiani, C., Serra, R., Villani, M., Kauffman, S.A., Colacci, A.: Cell interaction and diversity of emergent behaviors (submitted)
19. Socolar, J.E.S., Kauffman, S.A.: Scaling in ordered and critical random boolean networks. *Phys Rev. Lett.* 90 (2003)
20. Thomas, M., Cover, J.A.T.: Elements of information theory, 2nd edn. Wiley-Interscience, New York (2006)
21. Steuer, R., Kurths, J., Daub, C.O., Weise Selbig, J.: The mutual information: Detecting and evaluating dependencies between variables. *Bioinformatics* 18, S231–S240 (2002)
22. Schreiber, T.: Measuring information transfer. *Phys. Rev. Lett.* 85(2), 461–464 (2000)

Open Environment for 2d Lattice-Grain CA

Guillaume Cottenceau* and Dominique Désérable

INSA – Institut National des Sciences Appliquées,
Laboratoire de Génie Civil & Génie Mécanique,
20 Avenue des Buttes de Coësmes, 35043 Rennes, France
gcottenc@gmail.com, deserable@insa-rennes.fr

Abstract. An open cellular automata (CA) environment applied to the simulation of two-dimensional granular flows is presented herein. The CA belongs to the family of so-called “lattice-grain” (cellular) automata (LGrA) with one particle per cell. The time evolution is governed by a “request-exchange” synchronous mode which simulates a two-stage interaction-advection process. The transition rule follows a simple logic including three physical components: an external field, a set of kinematical exclusion rules and an inertial effect. After a short presentation of the CA logic, this paper describes the open user interface structured onto and emphasizes the versatility of the model.

Keywords: cellular automata, lattice-grain CA (LGrA), granular flow, user-friendly interface, open source.

1 Introduction

A thorough investigation into the behavior of granular matter is of major importance for scientific and industrial applications. The theoretical methods currently used to tackle these problems split up into the three distinct levels of: continuum models, particle dynamics and cellular automata. Cellular automata may capture the essence of physical phenomena resulting from elementary factors and make a suitable and powerful tool to catch the influence of the microscopic scale onto the macroscopic behavior of complex systems. Moreover, an important feature of cellular automata is their capability of handling complicated geometries and boundary conditions, where classical computational methods may fail or involve extra difficulty to model. Characterized by a high number of cells and a low transport of information, cellular automata can simulate a diversity of complex processes as encountered in dynamical systems [1,2]. Known as “lattice-gas” (cellular) automata (LGA) in hydrodynamics, they are an extreme simplification of molecular dynamics and have been widely developed over the last years for a better understanding of turbulence and stability phenomena in fluids, before being applied to the study of a larger diversity of physical systems [3,4,5,6,7,8]. Concerning granular media, there was a number of attempts which in turn make a relative simplification of granular dynamics and which are often

* INSA – Computer Science Department, until 2000.

known as “lattice-grain” (cellular) automata (LGrA); they have yielded some interesting results especially for hopper flows, flows around obstacles, segregation or stratification phenomena during free surface multiphase flows or the formation of density waves in channel flows [9,10,11,12,13,14]. A state of the art for lattice-grain models is given in [15] with references therein.

We focus on our LGrA described in detail in [16] and on the open environment associated with. This software is recognizable under the name “Grany-3” as a freeware [17,18]. Our model, derived from the analytical model of Litwiniszyn-Müllins [19,20], is purely “kinematical”, i.e. it ignores force and acceleration and only takes into account the notion of space occupancy. The transition rule is provided with three original features. Firstly, a two-stage “request-exchange” synchronous logic which simulates a physical interaction-advection process and activates all cells while ensuring a uniform operating mode; secondly, an “exclusion” principle based upon an elementary logic allowing the user to diversify the behavior of the “solid” phase; thirdly a synchronous “propagative” mode which simulates the propagation of the “void” phase: for brevity’s sake, this mode is not tackled in this paper. Beyond its general contribution to the simulation of multiphase granular flows, our model offers a high versatility allowing the user to simulate a diversity of systems by merely adjusting the initial and boundary conditions.

This paper aims to emphasize its versatility and to display the user-friendly interface of the free software associated with. Section 2 describes our LGrA logic, illustrated in our software presentation in Section 3 with a case study. We conclude in Section 4 by emphasizing the advantages that the researcher involved in studies on granular matter could benefit by using, or at least exploring what could bring him this open source software.

2 LGrA Logic

2.1 Topology of the Network and Space Occupancy

The automaton is constructed on the 6-*valent* grid. As for the “FHP” lattice-gas [3], this 2d topology offers the greatest number of symmetries for a regular network: herein it maximizes the number of *degrees of freedom* (or directions) for a displacement as well as the upper bound of the *coordination number* (the number of contacts of a particle with its vicinity). The concise notation “*ndf*” ($0 \leq n \leq 6$) will be used for a law with n degrees of freedom. Owing to the site-cell duality, a hexavalent site is the center of a hexagonal cell and every site is connected to its six neighbors. We denote by N_s the “size” or number of sites of the network. Since the graph is regular with degree 6, the number of links connecting a pair of adjacent sites is clearly $3N_s$.

The *space occupancy* principle allows one and only one particle per site, whether it is a solid, liquid or gaseous one, the term “particle” being a purely formal denotation. Multiphase flows are considered, where a phase ϕ_i , indexed in the set $N_\phi = \{1, 2, \dots, n_\phi\}$ for a system of n_ϕ phases, denotes a set of particles provided with identical properties. Note that a “phase” refers either to different

states of the same material or to different kinds of materials. A working feature of the automaton is that any cell of whatever content performs the same action synchronously, that leads to a uniform operating mode over the whole network.

2.2 A Two-Stage Interaction-Advection

The interaction-advection process is performed by a two-stage transition according to an original “request-exchange” mechanism. We will prefer this two-stage term to the “propagation-collision” one often used in the lattice-gas terminology in order to avoid confusion with a long-range propagative interaction. In the *request* stage, each cell autonomously performs a computation composed of a precalculation followed by a random choice. The result is a *potential* direction of displacement which becomes the direction of request. In the *exchange* stage, a test is performed for each link of the network in order to detect whether an agreement has been reached between the potential directions yielded by both adjacent sites.

Request-Exchange Synchronous Logic. It is useful to define a “request” graph as well as an “exchange” graph on the hexavalent lattice. The cell performs a request for displacement in one direction *at most*. A request is represented by an oriented arc, say $x \rightarrow x'$, from one site x to another site x' (Fig. 1a). Whenever two arcs $x \rightarrow x'$ and $x' \rightarrow x$ coexist, they are replaced by the edge $x \leftrightarrow x'$ in the exchange graph (Fig. 1b). The exchange graph is clearly a non-connected graph of isolated edges and where each component stands for a binary reaction. (The fact that each cell performs one request *at most* ensures thereby a conflict-free process. So whenever two particles compete with one another to fill a common site, arbitration is handled by the particle filling the addressed target.)

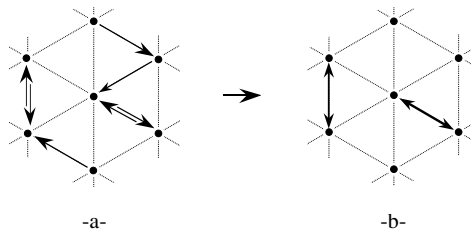


Fig. 1. The two-stage interaction-advection: (a) request graph and (b) exchange graph

Scheduling the Particle-Particle Exchanges. Requests are independently performed on each of the N_s sites whereas the particle-particle exchanges must be reviewed on each of the $3N_s$ links. In order to achieve the exchange stage without gap nor overlap a consistent scheduling is set up according to the, say, isotropic orientation $N \rightarrow S$, $SW \rightarrow NE$, $SE \rightarrow NW$ in Fig. 2a. This convention yields a $N-SW-SE$ input pattern as well as a $S-NE-NW$ output pattern uniformly

distributed on any site. In practice, the cell sends a copy of its own request to its $S-NE-NW$ neighbors while receiving a copy of the requests from its $N-SW-SE$ neighbors (Fig 2b). Then the occurrence of an exchange through the $N-SW-SE$ pattern is analysed. Finally the exchange with the N or SW or SE neighbor (if any) is performed.

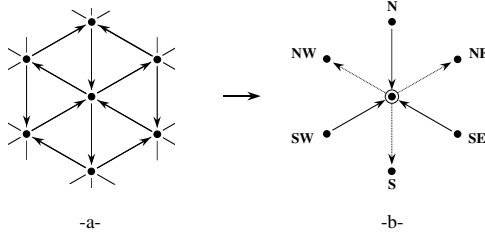


Fig. 2. (a) Orientation of the network and (b) exchange control

2.3 Phase Components and Interaction Law

The behavior of a *phase* in a multiphase system will be defined by three “physical” components: an external field, a set of exclusion rules and an inertial effect. It should be pointed out that it is not so much the autonomous behavior law of a given phase that must be taken into account but the *interaction* law with its local neighborhood. This fact is illustrated hereafter. In the sequel, a set $K = (0, 1, 2, 3, 4, 5)$ will be assigned to the six directions NE, N, NW, SW, S, SE .

Modeling the External Field. The action of an external field (in general the gravity) is modeled by assigning a 6-fold vector W to each phase. For simplicity, let us consider an isolated particle p_1 of phase ϕ_1 , immersed in a fluid of phase ϕ_2 and where $\rho_1 > \rho_2$ are the respective densities. To each phase ϕ_i is assigned a distribution of weights $W_i = (w_i^{(k)})_{k \in K}$ with non-negative integers, according to the orientations in Fig. 2. We adopt a “3df-3df” law with $W_1 = (0, 0, 0, 1, 1, 1)$, $W_2 = (1, 1, 1, 0, 0, 0)$ where the pattern $SW-S-SE$ means “downwards”. One request of ϕ_1 downwards will then result from W_1 and a swap may occur with a fluid particle below. Assume now p_1 immersed in a new fluid of phase ϕ'_2 with density $\rho'_2 > \rho_1$. Moving p_1 upwards requires *now* a distribution of the form $W_1 = (1, 1, 1, 0, 0, 0)$, $W'_2 = (0, 0, 0, 1, 1, 1)$.

From this observation, it follows that the external field is represented neither by W_1 nor by W_2 but by the set $\{W_1, W_2\}$. In a system of n_ϕ phases, the external field will be represented by a set $\{W_1, W_2, \dots, W_{n_\phi}\}$, where W_i is the distribution related to phase ϕ_i .

Exclusion Rules and Modes. The role of *exclusion rules* is firstly to ensure “kinematical compatibility” of the model. For instance, a usual rule consists in prohibiting the exchange of two solid particles lying in adjacent cells through their common link. More generally, their role is to differentiate the laws governing

the interactions between neighboring particles. They thus help to model some morphological (size, particle geometry...) or mechanical fuzzy notion (roughness, stiffness...) by introducing a binary parameter according to which a particle could either appear as “rough” or “smooth” in a monodisperse medium or “large” or “small” in a polydisperse medium... other physical scenarios are possible.

An *exclusion mode* is assigned to an exclusion rule and defined as follows.

- *pre-exclusion*: the rule is applied from the *request* stage onwards; to select a displacement direction, the cell is aware of the state of its six neighbors.
- *post-exclusion*: the application of the rule is postponed until just before the *exchange* stage; the cell performs a “blind” request with no consideration of the local vicinity.

As a matter of fact, the *pre* (resp. *post*) mode aims to affect a less (resp. more) frictional –or viscous– behavior to the phase which is assigned to. We define a set of three exclusion rules R_1, R_2, R_3 , namely “frontal”, “interstitial”, “constrictive”, more and more *coercive* when taken in this order (Fig. 3). Physically, assigning R_3 to phase ϕ implies R_2 which in turn implies R_1 to it. This sequence is shortly denoted by the implication $R_3(\phi) \Rightarrow R_2(\phi) \Rightarrow R_1(\phi)$. For consistency, a “neutral” rule R_0 will denote no exclusion for the assigned phase. The implementation of this exclusion logic is quite simple.

Inertial Effect. Modeling an “*inertial effect*” consists in saving the *memory* of the particle’s displacement direction at the previous timestep, to reintroduce it with a user-defined weight into the new weighted distribution for the current timestep. An *inertial* coefficient c_i , which takes on a positive integer value, is assigned to each phase ϕ_i and $c_i = 1$ means no inertial effect for this phase.

It may be asked what physical meaning could be attributed to the artifact giving inertia to the void. Whenever a high value is assigned to the “memory”

Rule	Rule name	Configuration
R_1	Frontal	
R_2	Interstitial	
R_3	Constrictive	

Fig. 3. Exclusion rules R_1, R_2, R_3 : applied to phase ϕ , a rule works when the neighboring sites occupied by ϕ' , or $\phi' - \phi''$, contain equally or more coercive phases. The barred arrow stands for the exclusion of the target site.

of the void phase, this tends to induce, when the void moves in a dense packing, an “indraught” to the particle located in the active direction. During a sequence of transitions, this void will move a row of grains one at a time but in the same direction. Although the row only moves at a rate of one particle per timestep, a sort of effect of void *propagation* may occur. Let us recall that a phase component is meaningful, only when embedded into the interaction law.

2.4 Time Evolution Equations

Modeling a Phase. For a given timestep, a cell contains a particle of phase ϕ_i ($i \in N_\phi$) characterized by the three following physical components.

- the action of an *external field*, depicted by a 6-fold vector

$$W_i = (w_i^{(k)})_{k \in K} \quad (1)$$

where weights $w_i^{(k)}$ are non-negative integers.

- the action of *exclusion rules*, precluding some direction or other depending on the state of the local vicinity and acting according to a *mode* from which the exclusion will be applied before (*pre-exclusion*) or after (*post-exclusion*) the request. This action is depicted by the 6-fold binary vector

$$\tilde{\mathcal{E}}_i = (\tilde{\varepsilon}_i^{(k)})_{k \in K} : \tilde{\varepsilon}_i^{(k)} = r_i \varepsilon_i^{(k)} + (1 - r_i) \quad (2)$$

where $\varepsilon_i^{(k)} = 0$ (or 1) whenever the site in direction k is excluded (or not) and where $r_i = 1$ (resp. 0) for a pre (resp. post) mode assigned to the phase.

- the action of a “memory”, or *inertial effect*, depicted by the 6-fold vector

$$\mathcal{M}_i = (\mu_i^{(k)})_{k \in K} \quad (3)$$

where $\mu_i^{(k)} = c_i$ if k was the displacement direction at the previous timestep and $\mu_i^{(k)} = 1$ otherwise. Coefficient c_i takes on positive integer values and $c_i = 1$ means no inertial effect for the phase.

Transition Algorithm. Prior to computing a request, a precalculation yields the corrected distribution

$$W_i^* = (w_i^{*(k)})_{k \in K} : w_i^{*(k)} = \mu_i^{(k)} \tilde{\varepsilon}_i^{(k)} w_i^{(k)} \quad (4)$$

and according to the exclusion mode, the probability of sending a request in direction k is given by

$$p_i^{*(k)} = \varepsilon_i^{(k)} \frac{w_i^{*(k)}}{\sum_K w_i^{*(k)}} \quad (5)$$

on condition that the sum of the corrected distribution be positive ($p_i^{*(k)} = 0$ otherwise). Direction k is selected at random by a pseudorandom sequence generated from a user-defined seed.

Let $(p_j^{*(k)})_{k \in K}$ be now the distribution of probabilities of the neighboring particle of phase ϕ_j in direction k and let X_k be the representative event of a displacement in direction k for the current particle. The probability of this event is finally

$$P(X_k) = p_i^{*(k)} p_j^{*(k+3 \bmod 6)} \quad (6)$$

according to the exchange protocol.

3 Software Presentation

Grany-3 is a C++ application implementing a simulator of our LGrA logic described in the previous section. It is organized around the following modules:

- a core engine which executes the two-stage request-exchange transition algorithm,
- a graphical interface which allows the user to define scenes from a vector-based geometry, to parameter the granular system from a user-friendly window set and to visualize the simulation steps,
- a file manager based on a text-based interface which saves the data of the scene as well as the simulation in progress for further external use.

3.1 A Case Study

We illustrate the presentation through a simple granular system simulating a silo emptying process. The silo is a container with a hopper at the bottom, provided with an outlet¹ through which bulk grain falls down.

A two-phase system $N_\phi = \{1, 2\}$ is considered, where ϕ_1 denotes the “grain” phase and ϕ_2 the “void” phase. The initial state is a silo fully filled with grains.

We adopt a “3df-3df” law with $W_1 = (0, 0, 0, 1, 2, 1)$, $W_2 = (1, 2, 1, 0, 0, 0)$ where the pattern $SW-S-SE$ of Fig 2b means “downwards”.

The frontal exclusion rule R_1 is assigned to ϕ_1 with the pre-exclusion mode. The neutral rule R_0 is assigned to ϕ_2 .

No inertial coefficient is assigned neither to ϕ_1 nor to ϕ_2 , namely, $c_1 = c_2 = 1$.

The walls of the container will be created by means of a graphical editor. The cells in the outlet region will play a special role of “void generator” by generating a void when a grain exits. The instantaneous flow rate is the number of voids generated per timestep.

This parameter set defines the local interaction law at the microscopic level. Macroscopically, it will involve a kind of last-in first-out pattern of “funnel flow”. In this example, we are not concerned by any physical interpretation of the observed phenomena: we are just interested by a presentation of the graphical interface.

¹ This term is preferable to “open source” in this context.

3.2 The Graphical Interface

Preparing a Scene. When starting the application, a *Scene Editor* should be open. After loading an existing scene, the scene editor activates the first **Borders** tab as shown in the front window of Fig. 4. Normal boundaries are drawn in blue, generating boundaries in red (the outlet generating “void” particles)². To create a new border file, to open an existing one or to modify the existing borders, the buttons on the right side should be used accordingly.

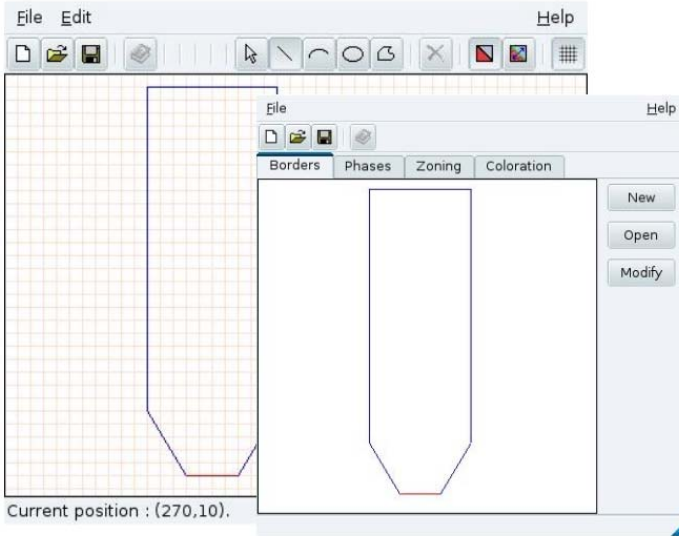


Fig. 4. The *Scene Editor* window, with its **Borders** tab activated. The *Border Editor* backwards, with its vector-based graphical tools displayed on the right of the toolbar.

When modifying the boundaries, a vector-based *Border Editor* is used as shown in Fig. 4 backwards. Available geometric forms (lines, arcs of ellipses, ellipses and polygons) are provided as buttons in the toolbar, and normal borders –or generating borders– can be switched from there too. A light pink grid is shown to facilitate the drawing of straight lines.

The second **Phases** tab of the scene editor will open a window to create the list of the two phases in the N_ϕ set. This main list is not displayed here. Phases will be edited with a **Properties** button and they can be set as **Propagative** and-or **Generated** by clicking an appropriate button. We create a first phase named **grain** and a second phase named **void** to which the **Generated** function will be assigned, that will involve a generation of voids in the red outlet region previously set up from the *Border Editor*. Neither the **grain** nor the **void** is defined as **Propagative**.

² We apologize for this coloured presentation in a black and white context.

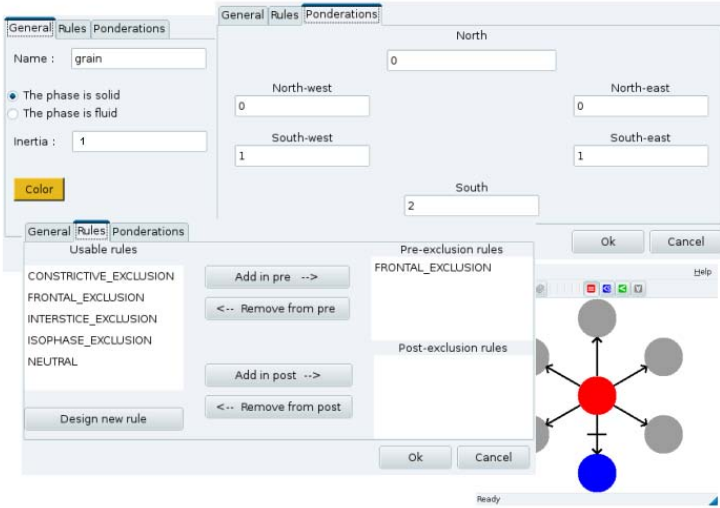


Fig. 5. The window environment to set up a selected phase. From back to front: the **General** window, the **Ponderations** window, the **Rules** window. A *Rule Editor* can be displayed to customize a new rule from the “Design new rule” button.

When the **grain** phase is selected in the **Phases** list, the **Properties** button opens a window with a **General** tab activated, as shown in the upper back window on the left side of Fig. 5. The **grain** is distinguished as a **solid** phase with no **Inertia** and an associated user-defined **Color**. The **Rules** tab displays a window with a list of standard or user-defined existing rules. The **FRONTAL_EXCLUSION** rule is selected in pre-exclusion by the “Add in pre” button. New rules can be customized with a *Rule Editor* activated by the “Design new rule” button and displayed on the right side in a window containing a 6-neighborhood template; since both frontal and neutral rules exist in the library, this tool will not be used in this example. Finally, the **Ponderations** tab displays a window to define the weighted vector W_1 . A similar environment will be activated by selecting the **void** phase in the **Phases** list.

Activating the third **Zoning** tab of the *Scene Editor* opens a *Zoning Editor* to create the initial state of a silo fully filled with grains. The back window in Fig. 6 shows a relative mix of phases specified by the sliders on the right, with 100% for the **grain** phase and 0% for the **void** phase. The “filling” is carried out from one of the geometric frames available in the toolbar. Note that the zoning rectangle does not need to fit perfectly into the frame of the container: it suffices to roughly cover it. The colour is meaningful: it is the colour chosen in Fig. 5 by the user for the **grain** phase.

The last **Coloration** tab of the *Scene Editor* shows an additional layered coloration in the front window. This artefact is not physically meaningful but very useful: it merely overlays the normal color of the grain phase, in order to better visually track the movement of individual grains during the simulation.

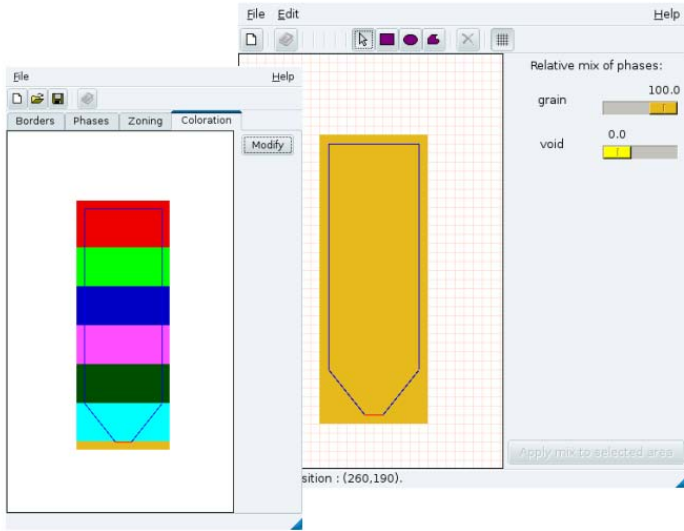


Fig. 6. The *Zoning Editor* window backwards, with its vector-based graphical tools on the toolbar to define various zoning regions. The *Scene Editor* in front, with its *Coloration* tab activated.

The scene is now ready, so we save it from the file manager toolbar of the *Scene Editor* as a “scene” file named `sil0.scene`.

Creating a Simulation. To create a simulation, the scene file `sil0.scene` should be reopened from the browser of a small dialog box, not shown herein. In this box, the user enters the three following parameters: the **Network size** of the cellular automata, the required **Number of steps** and the **seed** of the pseudorandom sequence.

Once the network size is known, the actual cells are generated to match the borders, phases, zonings and colorations of the scene as shown in the initial state of Fig. 7. The simulation toolbar is provided with a **Stop-Play-Pause** button set. The simulation starts when clicking on the **Play** button. Snapshots of the time evolution of the silo emptying process are displayed for $T = 0$, $T = 750$, $T = 1500$. From the toolbar, the user has a full control on the simulation steps and can save selected states from the file manager. Unless required otherwise, the simulation will stop once the number of steps is reached.

Network size and **seed** are two important parameters. The user can adjust the size of the model from coarse-grain (e.g. $16 \times 16 = 256$ cells) to fine-grain (e.g. $512 \times 512 = 262144$ cells) to study scale effects and to investigate scaling laws and critical phenomena for homogenization procedures. On the other hand, running several simulations with different seeds of the pseudorandom sequence allows the user to perform statistical investigations.

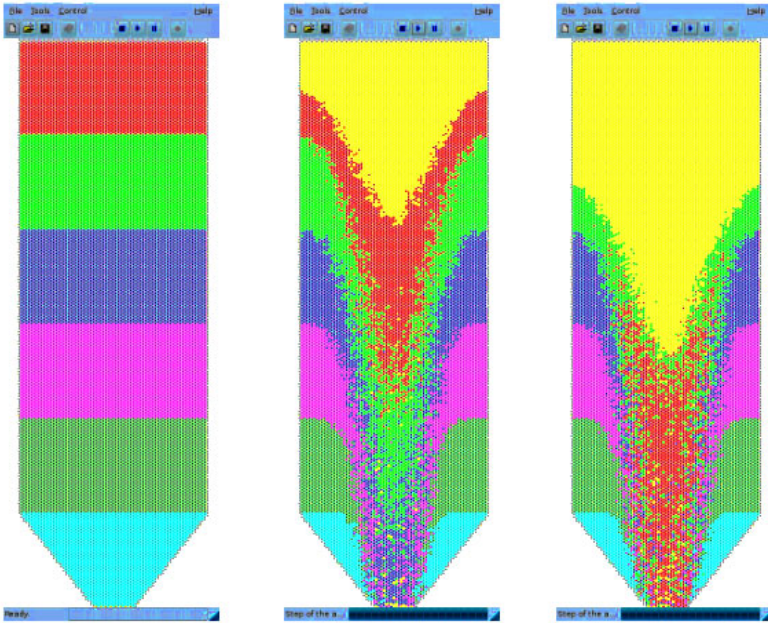


Fig. 7. Simulation of a silo emptying process. The toolbar displays the file manager tools to save data and the **Stop-Play-Pause** button set controlling the simulation. The cursor in the bottom displays the current step of the simulation in progress. The network contains 71289 cells. Snapshots at time $T = 0$, $T = 750$, $T = 1500$.

4 Conclusion

This software is very flexible and easy to use. In this case study, it is not difficult to modify a given parameter: for example, changing the angle of the hopper or the diameter of the outlet and observing their influence on the flow pattern could be readily performed, and so forth... The data produced by the file manager during a simulation process are available in a post-processing context for external use. In particular, flow patterns, flow rates, velocity profiles, density profiles, gradients, pressures, power laws and other macroscopic entities could be investigated. The aim of this paper was to present this open CA environment as a freeware that the researcher, involved in the study of the complex behavior of granular matter, may explore and customize for his own. Comparative studies with other computational methods like finite elements or granular dynamics appear as promising challenges which come within these objectives.

References

1. Wolfram, S.: Statistical Mechanics of Cellular Automata. Rev. Mod. Phys. 55, 601–644 (1983)
2. Kirkpatrick, S., Swendsen, R.H.: Statistical Mechanics and Disordered Systems. Comm. ACM 28(4), 363–373 (1985)

3. Frisch, U., Hasslacher, B., Pomeau, Y.: Lattice-Gas Automata for the Navier-Stokes Equation. *Phys. Rev. Lett.* 56(14), 1505–1508 (1986)
4. Wolfram, S.: Cellular Automata Fluids I: Basic Theory. *J. Stat. Phys.* 45, 471–526 (1986)
5. McNamara, G.R., Zanetti, G.: Use of the Boltzmann Equation to Simulate Lattice-Gas Automata. *Phys. Rev. Lett.* 61(20), 2332–2335 (1988)
6. Rothman, D.H., Zaleski, S.: *Lattice Gas Cellular Automata*. Cambridge University Press, Cambridge (1997)
7. Chopard, B., Droz, M.: *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, Cambridge (1998)
8. Talia, D., Sloot, P.: Cellular Automata: Promise and Prospects in Computational Science. Special issue of *Fut. Gen. Comp. Sys.* 16, 157–305 (1999)
9. Baxter, G.W., Behringer, R.P.: Cellular Automata Models of Granular Flow. *Phys. Rev. A* 42, 1017–1020 (1990)
10. Peng, G., Herrmann, H.J.: Density Waves of Granular Flow in a Pipe Using Lattice-Gas Automata. *Phys. Rev. E* 49, R1796–R1799 (1994)
11. Károlyi, A., Kertész, J., Havlin, S., Makse, H.A., Stanley, H.E.: Filling a Silo with a Mixture of Grains: Friction-Induced Segregation. *Europhys. Lett.* 44(3), 386–392 (1998)
12. Ktitarov, D.V., Wolf, D.E.: Stratification of Granular Matter in a Rotating Drum: Cellular Automaton Modelling. *Granular Matter* 1, 141–144 (1998)
13. Désérable, D., Masson, S., Martinez, J.: Influence of Exclusion Rules on Flow Patterns in a Lattice-Grain Model. In: Kishino, Y. (ed.) *Powders and Grains 2001*, Balkema, pp. 421–424 (2001)
14. Cisar, S.E., Ottino, J.M., Lueptow, R.M.: Geometric Effects of Mixing in 2D Granular Tumblers Using Discrete Models. *AIChE Journal* 53(5), 1151–1158 (2007)
15. Désérable, D., Dupont, P., Hellou, M., Kamali-Bernard, S.: Cellular Automata Models for Complex Matter. In: Malyshkin, V.E. (ed.) *PaCT 2007*. LNCS, vol. 4671, pp. 385–400. Springer, Heidelberg (2007)
16. Désérable, D.: A Versatile Two-Dimensional Cellular Automata Network for Granular Flow. *SIAM J. Applied Math.* 62(4), 1414–1436 (2002)
17. Cottenceau, G., Cruchant, S., Garcia, P., Le Guelvouit, G., Michard, X., Padioleau, Y., Zemali, Y.: Grany-3 Project: Design of a Cellular Automaton Simulator Dedicated to Granular Media. Technical Report, INSA Computer Science Department, Supervisors: Désérable, D., Rozé, L (1999)
18. Cottenceau, G.: Grany-3, <http://freshmeat.net/projects/grany3>
19. Litwinişzyn, J.: Application of the Equation of Stochastic Processes to Mechanics of Loose Bodies. *Archivum Mechaniki Stosowanej* 8(4), 393–411 (1956)
20. Müllins, W.W.: Stochastic Theory of Particle Flow under Gravity. *J. Appl. Phys.* 43, 665–678 (1972)

All-to-All Communication with CA Agents by Active Coloring and Acknowledging

Patrick Ediger and Rolf Hoffmann

Technische Universität Darmstadt
FB Informatik, FG Rechnerarchitektur
Hochschulstr. 10, 64289 Darmstadt, Germany
{ediger,hoffmann}@ra.informatik.tu-darmstadt.de

Abstract. We modeled a multi-agent system as a two-dimensional Cellular Automata and searched for a rule in order to solve the all-to-all communication task in shortest time. The rule contains two finite state machines (FSM) controlling the behavior of the uniform agents. The moving FSM controls the moving actions and the color FSM controls the changing of the cell's color. Colors are used for indirect communication. In addition the agents receive an acknowledgment whenever they meet and communicate successfully. The FSMs were evolved by a genetic algorithm. It could be shown that acknowledging and especially coloring increases the performance of the agents. Certain initial configurations cannot be solved without coloring. Even with coloring, symmetric configurations cannot be solved when the initial colors are the same.

Keywords: Cellular Automata, All-to-all Communication, Active Coloring, Evolving Behavior, Multi-Agent System.

1 Introduction

The general goal of this paper is to design multi-agent systems within the Cellular Automata (CA) model and to find the agents' rules that can solve a given global task. In this paper we investigate if and to which extent indirect communication via colors and acknowledging by a local feedback can lead to a better behavior.

The global task is all-to-all communication: Several agents are moving around in a two-dimensional CA grid with wrap-around in order to exchange their information among each other. Each agent initially has got one part of the mutually distributed information which can be exchanged when the agents meet in certain defined local patterns (communication situations). An agent system is considered to be *successful* when all agents have gathered the complete information. The communication situations (Fig. 1) are defined for patterns where two or more agents point to the same cell (the mediator).

The behavior of the agents is part of the CA rule and defined by finite state machines (FSM). The goal is to find optimal FSMs (needing the least number of steps for being successful) by evolving them with a Genetic Algorithm (GA). Note that the agents have to be optimized with respect to their moving

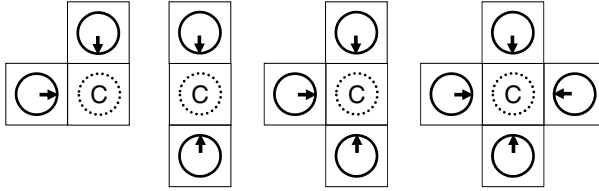


Fig. 1. Communication situations. Communication is done by exchanging the information between agents via a mediator cell C .

behavior. They should move in such a way that they meet each other with a high probability. The coloring is supposed to help the agents to take good movement decisions in order to meet other agents and exchange the information. Furthermore an acknowledging signal is available, giving the agents a feedback on whether information was gathered or not. More details of the CA model are given in Sec. 2.

In this contribution we pose the following questions: (1) Can the coloring of the cells, which is actively controlled by the agents, improve the all-to-all communication task? (2) How effective is an acknowledging feedback? (3) Are the agents successful on an arbitrary initial configuration?

All-to-all communication is a very common task in distributed computing. The problem's specification can depend on many fixed or dynamic varying parameters like the number and location of nodes, the number and location of processes, the number, users and properties of the communication channels and so on. All-to-all communication in multi-agent systems is related to multi-agent problems like finding a consensus [1], synchronizing oscillators, flocking theory or rendezvous in space [2], or in general to distributed algorithms with robots [3]. We have already studied the problem of all-to-all communication: In [4,5] the communication situations were defined without a mediator cell and no feedback or coloring was used. In [6] a simple form of coloring was used for indirect communication, but no direct feedback. In all former investigations the possible moving actions were limited. In this investigation we use a more powerful set of moving actions (see Sec. 2).

In former investigations [7] we have tried to find the best behavior for the *Creatures' Exploration Problem*, in which the creatures (agents) have the task to visit all empty cells in shortest time. The presented problem is related to it with respect to finding an optimal movement of the agents. Our research in general is also related to works like: evolving near optimal rules for cellular automata (CA) [8,9], finding out the center of gravity by marching pixels by evolutionary methods [10], modeling multi-agent systems in CA to simulate pedestrian movement [11] or traffic flow [12]. Modeling the behavior with a state machine with a restricted number of states and evaluation by enumerations was already undertaken in [13] and in [14].

The remainder of this paper is organized as follows. In Sec. 2 the CA modeling of the multi-agent system including the abilities of the agents is explained. The

method of evolving the agents' behavior and the problem sets are described in Sec. 3. In Sec. 4 the performance of the evolved agents is discussed and Sec. 5 concludes.

2 CA Modeling of the Multi-agent System

The whole system is modeled according to the cellular automata paradigm. It consists of an $n \times m$ grid of cells without borders (wrap-around, cyclic-boundary). We decided for the cyclic-boundary case because we know from former investigations that this case is much more difficult compared to the case with boundary. Each cell is able to model an agent or a free cell.

Neighborhood. All neighbors that might be accessed are within the Manhattan-distance of two (12 neighbors: N, E, S, W, NN, EE, SS, WW, NE, NW, SE, SW). A free cell uses only the NESW neighborhood, whereas an agent uses only the four neighbors in its moving direction.

Cell's State and Agent in General. The components of the cell's state are (*Type*, *Color*, *Direction*, *CommunicationVector*, *MoveControlState*, *ColorControlState*). *Type* defines the actual functionality (rule to be applied) of the cell, it distinguishes between an AGENT or a FREE cell. *Color* represents an information stored in the environment in order to speed-up the task. We will use either no color (for comparison) or two colors (0/1)¹. *Direction* *N, E, S, W* defines the moving direction of an agent. *CommunicationVector* stores the currently gathered information of an agent. *MoveControlState* and *ColorControlState* define the current state of the agents' controlling FSM (see below). The relevant state components depend on the type: If (*Type* = AGENT) all state components are relevant. All state components are visible for the neighbors, and all state components can be modified. If (*Type* = FREE) the components (*Type*, *Color*) are visible for the neighbors.

Modeling the movement from *P* to *Q* in CA requires two consistent rules based on the same information: (a) the cell *P* hosting the agent deletes the agent, and (b) the free cell *Q* copies the agent from *P*. If an agent is copied by a free cell, all the components except the color are copied. If no agent is moving to a free cell, the cell's state remains unchanged. This principle is described in more detail in 4.

An agent

- reads the information from the cell ahead (front cell) in order to detect an agent in front.
- reads the data of its neighbors in the moving direction (e. g., the neighbors are N, NN, NE, NW, if the moving direction is N) in order to detect a conflict 4.
- reads and writes the color on its own location.
- updates its control state according to the control algorithm.

¹ We have also experimented with more than two colors but the improvements were not as high as expected for this application.

If the front cell of an agent is occupied or a conflict occurs, the agent will stay on the current cell. A conflict occurs when two or more agents want to move to the same front cell (communication point, mediator)². Note that conflicts are necessary and useful to fulfill the task, because the conflicts match the communication situations.

A bit vector (*Communication Vector*) of length $k = \#agents$ is stored in each agent in order to accomplish the distribution of information. At the beginning the bits are set mutually exclusive (bit(i)=1 for agent(i)). When two, three or four agents form a communication situation, they exchange their information by simply OR-ing their bit vectors together. The task is successfully solved when the bit vectors of all agents obtain 11 ... 1.

Actions. An agent can perform a moving action $m \in \mathbf{M} = \{Sm, Rm, Lm, Bm, S, R, L, B\}$ and a coloring action $c \in \mathbf{C} = \{C0, C1\}$. The actions are

- *Sm*: straight ahead move conditionally
- *Rm/Lm/Bm*: turn right/left/back and move conditionally
- *S*: stay in the same direction and wait
- *R/L/B*: turn right/left/back and wait
- *C0*: clear color $g := 0$
- *C1*: set color $g := 1$

“Move conditionally” means that the agent will move if it is not hindered by other agents or a conflict.

Apart from the agent’s movement and the information exchange, an agent has indirect communication capabilities. Each cell of the environment contains a color which is either 0 or 1 and used as an input for the decision making process. The color can be seen as a tracing information like a “pheromone” left by other agents or even by the reading agent itself. In total one out of 16 actions can be selected, $mc \in \mathbf{M} \times \mathbf{C}$.

Behavior. The decision, which of the actions will be performed, depends on the behavior of the agent. The behavior (algorithm) of an agent is defined by a control automaton (FSM) (Fig. 2) which has a more complex structure compared to the one used before 7. The control automaton consists of a *move FSM* and a *color FSM*. Although it is possible to use only one FSM we decided for two separate specialized FSMs in order to reduce the overall complexity and the effort to evolve them. Output of the color FSM is a coloring action. Output of the move FSM is a moving action that is additionally checked for conformity (if the action tells “move” but a moving is not possible, then the agent has to wait).

The inputs of the control automaton are

- the type and direction of the neighbors in order to compute the move condition x : If (front cell == OBSTACLE \vee AGENT \vee CONFLICT) then $x = 0$ else $x = 1$.

² Alternatively the conflict detection can be realized by an arbitration logic 7 which is available in each cell. The arbitration logic evaluates the move requests coming from the agents and replies asynchronously by an acknowledge signal in the same clock cycle.

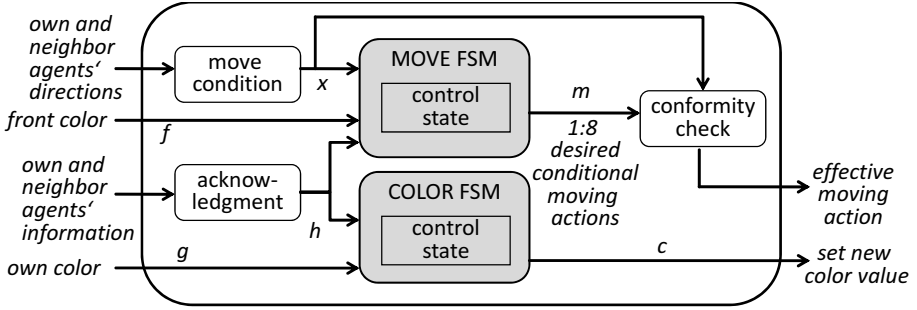


Fig. 2. The control automaton defining the behavior of the agent consists of two FSMs, one controls the movement and one controls the setting of the colors

- the color f of the front cell, used by the move FSM.
- the acknowledgment h that becomes 1, if the communication vector changes (amount of ones increases) when the agents are meeting, otherwise becomes 0. This information can be seen as a local feedback for a successful communication.
- the own color g , used by the color FSM.

An FSM is defined by a state transition table (Fig. 3) with current inputs $(f, h, x)/(g, h)$, current state s , next state s' and current output y . In order to keep the control automaton simple, we restrict the number of states and actions to a certain limit (see Sec. 3).

(a)				(b)																			
Inputs				Current State → Next State, Move Action								Inputs				Current State → Next State, Color Action							
f	h	x		0	1	2	3	4	5	6	7	g	h	0	1	2	3	4	5	6	7		
0	0	0		3,S	3,Sm	4,Sm	0,Lm	3,Lm	3,S	0,B	3,B	0	0	4,C1	2,C0	6,C1	0,C1	6,C1	0,C0	4,C1	1,C0		
0	0	1		4,Sm	3,Sm	7,Sm	6,L	6,S	1,S	5,Sm	7,Sm	0	1	3,C0	1,C1	5,C0	7,C0	5,C0	4,C1	4,C0	7,C1		
0	1	0		0,L	7,L	0,L	1,B	2,Lm	5,R	7,Sm	4,Lm	1	0	2,C1	6,C1	5,C0	0,C0	6,C0	3,C1	6,C1	7,C1		
0	1	1		1,L	0,Sm	5,S	4,S	1,R	2,Sm	5,B	5,B	1	1	2,C1	1,C1	3,C0	1,C1	3,C1	2,C1	6,C1	5,C1		
1	0	0		6,S	7,Sm	3,Sm	0,B	5,Lm	3,B	1,Lm	3,Rm												
1	0	1		7,B	4,B	3,B	3,Sm	7,Sm	3,Sm	6,Sm	3,Sm												
1	1	0		5,Rm	4,Lm	1,R	7,Bm	6,Sm	3,B	0,Bm	3,Bm												
1	1	1		3,L	4,Sm	3,S	4,S	7,R	5,S	4,B	5,B												

Fig. 3. Example of the state tables for (a) the move automaton and (b) the color automaton. The best found behavior is shown here (see Sec. 4). The number of states is restricted to 8 for both FSMs. f stands for *front color*, g for *own color*.

Cell Rule. The cell structure (Fig. 4) consists of the cell’s state components, the control automaton and a function r . The complete cell rule, realized by the function r and the control automaton, can be informally described as follows:

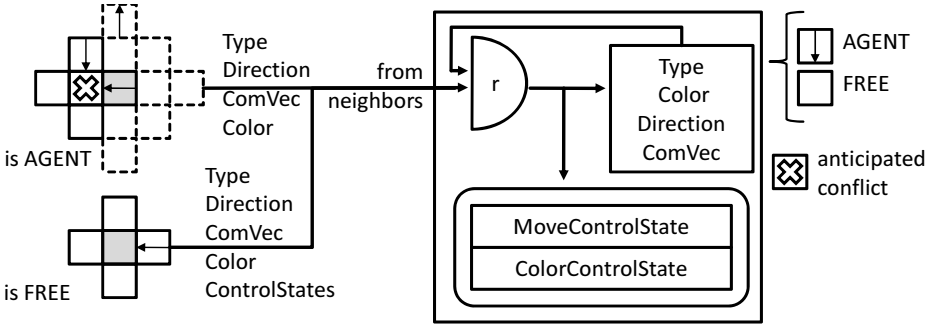


Fig. 4. The structure of a cell. The control automaton is embedded in the cell structure. The function r computes the new state with the inputs of the neighbors and the own state. *ComVec* is an abbreviation for *CommunicationVector*.

- If ($Type = AGENT$), detect whether the movement to the front cell is possible. (a) If it is possible, change $Type$ to FREE, use inputs g and $h = 0$ to perform the state transition of the color FSM and update $Color$ according to the output of the color FSM. (b) If it is not possible, detect communication situations and update *CommunicationVector*, perform the state transitions of both FSMs using inputs f, g, h, x and update $ColorControlState$, $MoveControlState$ as well as $Color$ and $Direction$ according to the outputs of the FSMs. Note that a neighborhood with Manhattan distance of 2 is needed here.
- If ($Type = FREE$), calculate whether exactly one neighboring cell with $Type=AGENT$ pointing to the own cell exists. If such an agent exists, copy the $MoveControlState$, $Direction$ and use it in the own move FSM to determine (in the own cell) the next state, and the turning decision of the move FSM. Copy also the $ColorControlState$ to perform the state transition in the own cell (but do not update $Color$). Then update *CommunicationVector* by copying it from the agent cell and finally change $Type$ to AGENT.

3 Investigations

The goal is to find the optimal³ agents' behavior, which is defined by a combination of a move FSM and a color FSM, for all possible initial configurations of the CA grid. The method is to evolve the FSMs by a Genetic Algorithm (GA) evaluating the fitness through simulation of the CA. Different initial configurations vary in the size of the grid, the number of agents in it, and their position and direction.

In order to find out the effects of the coloring ability and the acknowledging feedback, we performed the search for a near optimal behavior in four models, which all are special cases of the general model described in Sec. 2:

³ Since we cannot prove or reach optimality, we use the term “optimal” in the sense of “near optimal”.

- A. agents with coloring ability and with acknowledging
- B. agents with coloring ability and without acknowledging
- C. agents without coloring ability and with acknowledging
- D. agents without coloring ability and without acknowledging

At first we used a *Training Set* of 10 initial grids of size 33×33 with 16 randomly placed agents. This set was taken over from former investigations [6]. From former investigations we know that a number of agents between 8 and 64 in this grid size leads to a sufficient number of communication situations. In addition we used an *Extended Set* (Sec. 4) in order to test for robustness.

Fitness Function. The quality of a solution is given by the fitness function $F = 10^5(\#a - a) + 10^4(1 - c) + t_c$. It reflects three aspects:

1. The number of agents a (maximum $\#a$) which have gathered the complete information. If an agent has gathered the complete information it is *informed*. If all agents are informed, we characterize the agent system respectively the algorithm as *successful*. If the agents are successful on all given initial configurations then we characterize the agent system respectively the algorithm as *completely successful*.
2. The algorithm should perform at least one communication ($c = 0/1$).
3. The number of steps in the CA simulation to reach successfulness. We will call this value also *communication time* (t_c).

For successful algorithms, the relation *Fitness = communication time = steps* holds. Note that a lower fitness value is better. The fitness value for the Training Set is the average of the fitness values of all configurations.

Genetic Algorithm. The search space for different behaviors is very large, because the number of FSMs that can be coded with $\#s$ states, $\#x$ possible input values and $\#y$ possible output values is $K = (\#s\#y)^{\#s\#x}$. Due to the exponential growth, the number of states was restricted to $\#s = 8$, which is sufficient to get feasible solutions as former investigations showed [6]. In the cases A and B a pair of two FSMs (move FSM/color FSM) has to be evolved, whereas in the cases C and D only the move FSM has to be evolved. Exact values of the search space for each case are given in Table II.

A state table can be coded by a concatenation of all pairs (s', y) . s' is the next state, y the output to a given combination of current state and input. This string defines a genome of one individual, a possible solution. In the cases A and B, the concatenation of the strings of the two automata define the genome. An Island Model Genetic Algorithm is used to evolve the pairs of FSMs. P populations with N individuals are updated in each generation (optimization iteration). During each iteration, M children are produced in each population. The union of the current N individuals and the M children are sorted according to their fitness and the N best are selected to form the next population. The offspring is produced as follows:

1. (GET PARENTS) Two parents are chosen for each population. Each parent is chosen from the own population with a probability of p_1 and from an arbitrary other population with the probability of $(1 - p_1)$.

2. (CROSSOVER) Each new component (s'_i, y_i) of the genome string is taken from either the first parent or the second parent with a probability of 50%. This means that the tuple (next state, output) for the position $i=(input, state)$ is inherited from either of the parents.
3. (MUTATION) Each component (s'_i, y_i) is afterwards mutated with a probability of p_2 . Thereby the next state and the output at position i are changed to a random value.

The probabilities were set to $p_1 = 98\%$ and $p_2 = 0.9\%$. In each case six independent runs with 10,000 GA generations were performed and the results were averaged where it makes sense. With $P = 5$, $N = 100$ and $M = 10$ per island per generation this leads to 500,000 tested behaviors in all four cases. In total 5,000,000 simulations (10 environments \cdot 500,000) were performed. In all computations the simulation of the CA is by far the most time consuming part of the calculation. Note that, because the solutions for case D are included in the ones for case C, they can be inserted at any time into the GA process for case C by replacing another solution. The same holds for the cases B and A.

4 Results

Completely successful behaviors could be evolved for the Training Set in the cases A-D. The best results were achieved for case A (Tab. 1, Fig. 3). The performance can be improved significantly by using colors. The usage of the acknowledgment h leads to a smaller improvement. The acknowledging effect is lower than the coloring effect because, compared to the number of color changes, it is relatively infrequent that two agents exchange a new information.

Table 1. The best fitness values of all evolved behaviors (averaged over six runs). “n.a.” means that some configurations are not solvable in principle.

CASE	COLOR	ACKNOWLEDGMENT	SEARCH SPACE K	FITNESS ON TRAINING SET	SUCCESS ON EXTENDED SET
A	yes	yes	$(64^{64})(16^{32}) \approx 1.34 \cdot 10^{154}$	202.8	yes
B	yes	no	$(64^{32})(16^{16}) \approx 1.16 \cdot 10^{77}$	211.7	not evolved
C	no	yes	$(64^{32}) \approx 6.28 \cdot 10^{57}$	366.7	n.a.
D	no	no	$(64^{16}) \approx 7.92 \cdot 10^{28}$	395.1	n.a.

Another reason for using the coloring ability is that, for some specific configurations, a solution without colors is not possible. This is the case when all agents initially have the same direction and there is enough free space in between them; then they will never meet in communication situations. This happens because all agents start in the same control state and always receive the same inputs. Thus all agents move and turn in a synchronized way (Fig. 5(a)). By the usage of colors, this problem can be overcome in some cases, because then the inputs of the different agents vary depending on the colors that were set by other agents,

and as a consequence they do not behave synchronized and can possibly meet in communication situations (Fig. 5(b)).

We designed an *Extended Set* of configurations by adding some manually designed configurations (e. g., equally distributed in a row or in diagonal, similar to Fig. 5(a)) which can only be solved with colors. We evolved a behavior for the case A and found completely successful agents (Tab. 1).

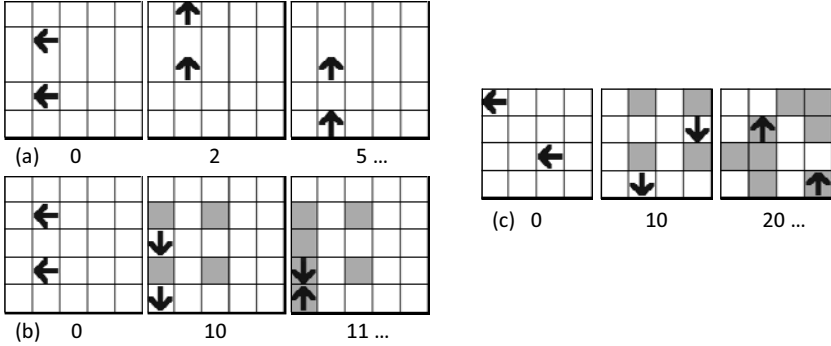


Fig. 5. Simulation snapshots of 5×5 and 4×4 configurations with two agents: (a) agents moving and turning in a synchronized way cannot solve this problem, (b) agents setting colors receive a different input at step 10 in this example, and thus act differently afterwards, (c) due to the equal distances through the wrap-around, there will be a synchronized behavior despite coloring.

Nevertheless there is still a third type of configurations that can not even be solved by agents with coloring ability. This is the case when there is not only enough space between the agents but additionally the environment looks exactly the same (symmetric) from each agent’s perspective, i. e., all distances are equal (Fig. 5(c)). There are several ways to solve this problem. Generally it is necessary to introduce a certain inhomogeneity. This could be done by (a) initially set different colors, (b) start agents in different initial control states, (c) use randomness during the decision process of the agent, (d) insert obstacles in the configuration, (e) use agents with different behaviors or (f) use asynchronous updating. Solution (a) was tested. The best FSM combination was simulated on an additional set of initial configurations with equal distances between the agents and initially set colors ($c = 1$) in the cells in front of half of the agents. Although it was not evolved for these configurations, it was completely successful.

The FSM strategy (Fig. 3) of the best evolved agents on the Extended Set seems to be straight forward. They use the ability to color the cells in order to mark “streets”. All agents will follow the course of the main streets when reaching them. Thus the possibility of meeting each other in communication situations is very high. Furthermore, the agents prefer to walk on horizontal or vertical “streets”. Since the neighborhood is a von Neumann neighborhood (NESW),

a diagonal way could only be walked in a stair-like manner. Then this “diagonal street” would occupy more cells and the probability of a meeting would be lower. The simulation sequence for one of the initial configurations (Fig. 6) shows that the cells that are most visited correspond to the cells which were colored.

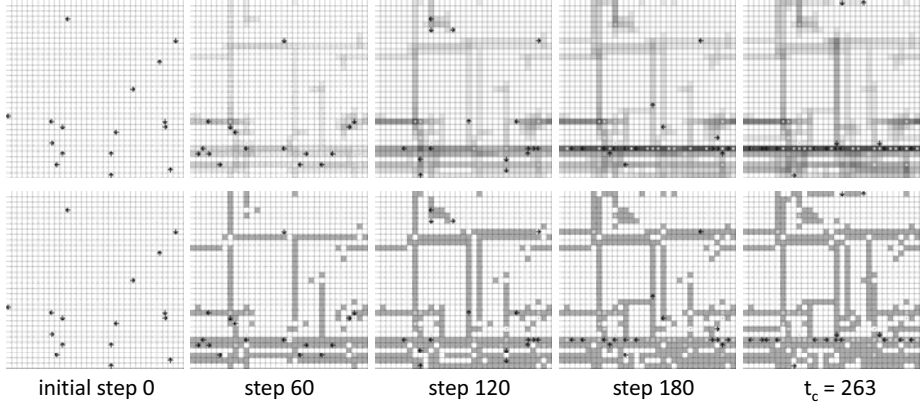


Fig. 6. Patterns of the visited cells (first line) and the colors (second line) for the best behavior during the simulation of one configuration. The agents are building a network of streets on which they move in order to communicate successfully.

5 Conclusion

The all-to-all communication task for a multi-agent system in a CA was studied applying indirect communication and acknowledging. The behavior of the agents was controlled by two FSMs: The move FSM controls the moving actions and the color FSM controls the colors. The FSMs were evolved with a GA. The best found behavior uses the coloring ability and the acknowledging. In comparison to systems without these abilities, they solved the task on the *Training Set* of initial configurations almost twice as fast.

It was recognized that without coloring certain initial configurations cannot be solved because of the uniform behavior and synchronous updating. Thus, an additional search (by GA) on an *Extended Set* of configurations containing these special cases was performed. The resulting behaviors showed to be more robust. But still there is another type of configurations, which even with the coloring ability cannot be solved. A few methods to overcome this problem were proposed, e. g. setting colors initially proved to be applicable.

In future investigations we will use more complex agents with extended communication and acknowledging abilities. For this purpose, the genetic algorithm shall be revised, too, as the complexity and the search space grow fast. We also want to examine systematically inhomogeneous systems that can guarantee to be successful on arbitrary initial configurations.

References

1. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE* 95, 215–233 (2007)
2. Lin, J., Morse, A.S., Anderson, B.D.O.: The Multi-Agent Rendezvous Problem. An Extended Summary. In: *Cooperative Control*. LNCS, vol. 309, pp. 257–289. Springer, Heidelberg (2005)
3. Principe, G., Santoro, N.: Distributed algorithms for autonomous mobile robots. In: *4th IFIP Int. Conf. on TCS*. IFIP, vol. 209, pp. 47–62. Springer, Heidelberg (2006)
4. Ediger, P., Hoffmann, R.: Optimizing the creature’s rule for all-to-all communication. In: *EPSRC Workshop Automata-2008. Theory and Applications of Cellular Automata*, Bristol, UK, June 12–14, pp. 398–410 (2008)
5. Hoffmann, R., Ediger, P.: Evolving Multi-creature Systems for All-to-All Communication. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 550–554. Springer, Heidelberg (2008)
6. Ediger, P., Hoffmann, R.: Solving All-to-All Communication with CA Agents More Effectively with Flags. In: Malyshkin, V. (ed.) *PACT 2009*. LNCS, vol. 5698, pp. 182–193. Springer, Heidelberg (2009)
7. Halbach, M., Hoffmann, R., Both, L.: Optimal 6-state algorithms for the behavior of several moving creatures. In: Yacoubi, S.E., Chopard, B., Bandini, S. (eds.) *ACRI 2006*. LNCS, vol. 4173, pp. 571–581. Springer, Heidelberg (2006)
8. Sipper, M.: *Evolution of Parallel Cellular Machines, The Cellular Programming Approach*. LNCS, vol. 1194. Springer, Heidelberg (1997)
9. Sipper, M., Tomassini, M.: Computation in artificially evolved, non-uniform cellular automata. *Theor. Comput. Sci.* 217(1), 81–98 (1999)
10. Komann, M., Mainka, A., Fey, D.: Comparison of evolving uniform, non-uniform cellular automaton, and genetic programming for centroid detection with hardware agents. In: Malyshkin, V.E. (ed.) *PaCT 2007*. LNCS, vol. 4671, pp. 432–441. Springer, Heidelberg (2007)
11. Dijkstra, J., Jessurun, J., Timmermans, H.J.P.: A multi-agent cellular automata model of pedestrian movement. In: Schreckenberg, M., Sharma, S.D. (eds.) *Pedestrian and Evacuation Dynamics*, pp. 173–181. Springer, Heidelberg (2001)
12. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. *J. de Physique* 2, 2221 (1992)
13. Mesot, B., Sanchez, E., Peña, C.A., Perez-Uribe, A.: SOS++: Finding smart behaviors using learning and evolution. In: Standish, R., Bedau, M., Abbass, H. (eds.) *Artificial Life VIII: The 8th International Conference on Artificial Life*, pp. 264–273. MIT Press, Cambridge (2002)
14. Halbach, M.: *Algorithmen und Hardwarearchitekturen zur optimierten Aufzählung von Automaten und deren Einsatz bei der Simulation künstlicher Kreaturen*. PhD thesis, Technische Universität Darmstadt (2008)

The Sandpile Model: Parallelization of Efficient Algorithms for Systems with Shared Memory

Sebastian Frehmel

IKS Vollmar, Karlsruhe Institute of Technology (KIT), Germany
sebastian.frehmel@isi.fraunhofer.de

Abstract. Having resisted successful parallelization so far, this paper shows how the sandpile cellular automaton can be efficiently parallelized on multicore computers with shared memory. New algorithms for both the sequential and parallel case are being introduced. Implementations in Java show a good speed-up which increases with the grid size.

1 Introduction

With the common availability of multicore computers with shared memory the wish arose to finally be able to parallelize the sandpile model on those machines. Previous attempts in parallelizing this model at the IKS and elsewhere failed predominantly applying message passing parallel computers with MPI. The cause for this failure is found in the inability to use the technique of domain decomposition as only very few cells need updating in a global cellular automaton step. Furthermore the overhead of communication between nodes of a parallel computer is too slow compared to the simple calculation of the sandpile model.

As a result, previous work focused on improving sequential versions. Although significant improvements for the sequential case had been given and proven correct, e.g. by Walter and Worsch [3], these solutions still consume a large amount of time until computations are completed. This is especially the case for large grid sizes of the sandpile cellular automata (CA) which are of special interest when analyzing the model. The special characteristic of the model, self-organized criticality, can then be observed best as it is shown more detailed.

The solutions stated in this paper use the same main idea as those existing solutions which only take into account the smallest possible amount of cells which are subject to be updated in the next CA step (as for example in [3]). Yet, the algorithms introduced here will be different.

A big obstacle when it comes to parallelizing the sandpile model is the big difference between the very short and simple process of calculation compared to a high demand and usage of memory. This poses a problem as the speed of main memory accesses did not develop as fast as the CPU speed in the last years and therefore creates a bottleneck. Therefore the layout of the used data structures in memory has to be especially optimized with respect to caches.

This paper is structured as follows: In Section 2 a quick review of the definition of CA followed by the explanation of the sandpile model as well as the

employed notions and symbols is given. Section 3 then describes two new sequential algorithms before Section 4 draws attention to the actual topic of this paper, the parallelization. In that section the concept and implementation are explained in detail focusing on efficient means of hardware locking which forms a central performance issue. The results of the parallelization process can be found in Section 5. The paper concludes with an outlook in Section 6.

2 Basics

2.1 Cellular Automata

We assume the reader is familiar with the concept of synchronous and asynchronous cellular automata. The following notations are used:

A d -dimensional cellular automaton A is a 5-tuple $(\mathbb{Z}^d, Q, N, \delta, q_0)$. \mathbb{Z}^d being a regular d -dimensional grid representing the structure of the CA. Q is a finite set of cell states and N a finite set of neighborhood indexes represented as coordinate differences, $\mathbf{0}$ not included. $x + N = \{x + n \mid n \in N\}$ stands for the set which contains all cells that are neighbors of x .

$\delta : Q^n \mapsto Q$ is the local transition function of the CA and configurations of A are denoted as mappings $c : \mathbb{Z}^d \mapsto Q$ with c_x being short for the state of cell x in configuration c .

2.2 Sandpile Model

The sandpile model was first introduced by Bak, Tang and Wiesenfeld in 1987 and continued in further detail in 1988 [1].

It is originally defined as a two-dimensional CA with von-Neumann neighborhood of radius 1. Then $Q = \{0, 1, 2, \dots, 7\}$ is the finite set of states.

Each cell can hold a certain amount $q \in Q$ of grains of sand. A cell is said to be *critical* if it contains four or more grains. Then the local transition rule is best described as follows:

- If a cell is critical ($q \geq 4$) it takes four grains from itself and distributes one of those to each neighboring cell. This action is called “firing”.
- If a cell is not critical ($q < 4$) the cell by itself remains passive. It can receive grains of sand from critical neighbors and possibly become critical itself.

If there are no critical cells left in a configuration, this configuration is said to be *stable*. Once having reached a stable configuration one new grain of sand is added to a randomly chosen cell of the grid until one cell becomes critical. Should a cell fire, neighboring cells may also become critical and fire. This chain reaction is called an *avalanche*. Once the sandpile CA has reached the state of self-organized criticality avalanche sizes grow rapidly.

The sandpile CA uses bounded grids only. With bounded grids cells at an edge can lose grains of sand. This behaviour is necessary to allow for relaxation and guaranteedly reaching a stable configuration.

It is known that the outcome of an avalanche is independent from its actual order of cell visits. This is equivalent to perceiving the sandpile automaton as an asynchronous CA. This is very important for the solutions provided with this paper. The number of global steps of a synchronous sandpile CA is the only information which cannot be obtained from an asynchronous solution.

All computational experiments for measuring the speed of implementations had the following structure: The grid was a square of size $n \times n$. In the initial configuration all cells were in state 0. For the simulation $2.175 * n^2$ grains were added to the grid expecting self-organized criticality to consolidate at approximately $2.13 * n^2$ throws.

3 Sequential Solutions

In contrast to the paper by Walter and Worsch [3] which makes use of two grids that are swapped every time one of the two used update sets is empty, we use only one grid in the two new sequential algorithms.

Both of them employ stacks as a central data structure instead of lists. By using stacks processing cells will be more affine to caches as cells which have been pushed onto the stack will be used again very soon afterwards. This is contrary to lists in which an element is always enqueued at the end and therefore won't be processed again quickly.

Similarly to the solution by Walter and Worsch the first sequential algorithm (see Algo. [1]) uses two update data structures. These structures are stacks in this case, called A and A' . The algorithm is quite easy to understand. The inner while loop processes the elements in the currently active stack A (lines 5-14), applying the transition function to it. Cells which have become critical again are pushed onto the other stack A' (line 11). If stack A is empty both stacks change their roles (line 16) and the trigger process starts anew with the new stack A . Should this stack also be empty new grains of sand are added to the grid until a cell becomes critical again. This algorithm works synchronously and hence even the number of global steps needed for an avalanche is available.

Taking into account the fact that a sandpile CA can also be realized using asynchronous updating, Algo. [1] can be simplified. Lines 1-4 are then equivalent to Algo. [1] and the continuation can be seen in Algo. [2]. Now only one stack and one while loop are necessary and this loop will only be left when a stable configuration is reached and a new grain of sand is to be added. As a result, this algorithm uses asynchronous updating.

The particularity with this solution is the possibility of a cell having a state greater than 7. This situation may occur if a cell repeatedly is a neighbor of a critical cell but due to the stack won't be accessed immediately as calculation progresses into other directions. In Algo. [2] the inner while loop (line 7) deals with that problem by triggering a cell as long as the cell remains in a critical state.

An important necessity which is shown by method `pushUnique` in both algorithms is a check for the "availability" of a cell. Once a cell is pushed onto the stack it won't be pushed again until it has been popped off the stack. This is realized using one bit in the memory of each cell (see Sect. [4.3]).

Algorithm 1. Sequential algorithm with two stacks and one grid

```

1 forall Grain  $g \in \text{grains}$  do
2    $z \leftarrow \text{cell hit by } g$ 
3   if  $\text{increment}(c_z) \geq 4$  then
4      $A.\text{push}(z)$ 
5     while  $A.\text{containsElement}()$  do
6       while  $A.\text{containsElement}()$  do
7          $x \leftarrow A.\text{pop}()$ 
8          $c_x \leftarrow c_x - 4$ 
9         forall  $y \in x + N$  do
10          if  $\text{increment}(c_y) \geq 4$  then
11             $A'.\text{pushUnique}(y)$ 
12          end
13        end
14      end
15       $A.\text{reset}()$ 
16       $A \leftrightarrow A'$ 
17    end
18  end
19 end

```

Algorithm 2. Sequential algorithm with one stack and one grid (excerpt)

```

5 while  $A.\text{containsElement}()$  do
6    $x \leftarrow A.\text{pop}()$ 
7   while  $c_x \geq 4$  do
8      $c_x \leftarrow c_x - 4$ 
9     forall  $y \in x + N$  do
10      if  $\text{increment}(c_y) \geq 4$  then
11         $A.\text{pushUnique}(y)$ 
12      end
13    end
14  end
15 end

```

4 Parallel Solution

4.1 Cache Optimization

We assume the reader is familiar with cache layouts and the notion of a cache line. With a realization of the sandpile grid as a one-dimensional array allocated in one chunk in memory, accessing an element of this array (i.e. a cell) will with a high probability have the neighboring cells to its left and right be in the same cache line as well. Accessing the top and bottom cells will result in cache misses. One goal should be an optimized access so that all neighboring cells and the cell itself are situated in one cache line.

In order to achieve this, the cells are not stored in the standard row major order. Instead, if k denotes the number of cells that fit into one cache line, roughly speaking the following approach is used: Square blocks of $\sqrt{k} \times \sqrt{k}$ cells are stored in column major order in a block of memory that fits exactly into one cache line. This optimization now allows many neighborhoods of a cell to be

within only a few array elements instead of being stored at an array index apart by as many elements as the CA is wide.

With a usual cache line size of 64 bytes 56.25% of all neighborhoods are now situated entirely in one block, leading to one initial cache miss only, 37.5% have one cell overlapping into another block, leading to two cache misses in total and only 6.25% have again two cells in other blocks. Theoretically, the cache hit rate rises from $\approx 40\%$ in the unoptimized case to $\approx 70\%$ in the optimized case. This optimization will be evaluated in Sect. 5.

4.2 Concept for Parallelization

Generally speaking our parallel program uses threads, each of which works like the sequential solution with one stack (see Algo. 2), and an efficient thread handler offering waiting threads to active ones which pass on work to those threads. Therefore, the algorithm of the threads has to be adapted slightly in order to function in a concurrent environment. Access to a cell of the grid has to be done exclusively in order to prevent race conditions.

The central data structure of the thread handler is a ring list consisting of queues with elements being threads (see Fig. 1). Before the threads start working every thread but one is enqueued into such a queue. The remaining thread starts to add grains to the grid. Once a cell fires and further neighboring cells become critical those cells are passed on to waiting threads in one of the queues. This activation removes the thread from a queue. If a thread finishes working, i.e. not having

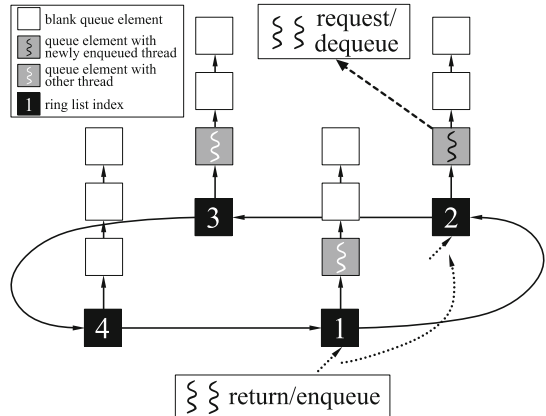


Fig. 1. The ring list of thread queues

a critical cell to trigger in its stack, it returns to the thread handler and is enqueued to a queue in the ring list in a round-robin fashion. Dequeue operations always try to take place at the queue which last has had an enqueue operation as then the probability of finding a waiting thread quickly will be higher. A sketch of the parallel algorithm can be found in Algo. 3. There a class `threadHandler` and the thread `workerThread` represent the elements mentioned above. The requested number of threads is started in parallel (line 43) before the first grain is thrown. This is done in function `throwGrain` (line 12). If a cell becomes critical, a thread is dequeued from the ring list of thread queues (member `tqr`) mentioned earlier. This thread then gets assigned the task to trigger this cell (line 16).

Algorithm 3. Parallel algorithm for parallel stack solution (lines 10 and 34 are simplifications of the real implementation)

```

1 class ThreadHandler()
2   var: ThreadQueueRing tqr
3   function enq(workerThread wt)
4     tqr.enqToNextQueue(wt)
5     if all threads have returned then
6       throwGrain()
7     end
8   end
9   function deq()
10    return tqr.deqFromCurrentQueue()
11  end
12  function throwGrain()
13    for Grain g ∈ grains do
14      z ← cell hit by g
15      cz ← cz + 1
16      if cz = 4 then
17        deq().startWith ← z
18      end
19    end
20  end
21 end
22 thread workerThread()
23   var: Cell startWith
24   repeat
25     wait until startWith gets assigned
26     stack.push(startWith)
27     while stack.containsElement() do
28       x ← stack.pop()
29       incrementCount ←  $\lfloor \frac{c_x}{4} \rfloor$ 
30       cx ← cx - 4 * incrementCount
31       forall y ∈ getCacheOptimizedNeighbors(x) do
32         cy ← cy + incrementCount
33         if cy ≥ 4 ∧ ThreadHandler.isThreadAvailable() then
34           ThreadHandler.deq().startWith ← y
35         else if cy ≥ 4 then
36           stack.pushUnique(y)
37         end
38       end
39     end
40     ThreadHandler.enq(self)
41   forever
42 end
43 runParallel (clone(workerThread, #Threads))
44 ThreadHandler.throwGrain()

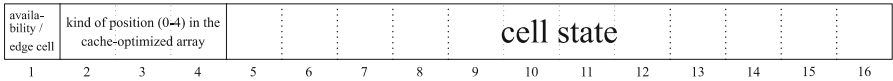
```

In the thread code, every thread waits for a cell to be triggered (line 25). Should a cell have been pushed onto the stack, then according to the sequential solution, this cell is triggered as long as it remains critical. In order to avoid a loop leading to more locks the number of overall trigger actions is calculated (line 29). After that, every neighboring cell is incremented atomically by that amount. If the neighboring cell became critical the currently running thread now has to check whether there are threads waiting in the thread handler or not. If there are threads, then work is passed on to a new thread by assigning it the currently processed neighboring cell (line 34). Shouldn't there be any waiting threads available, the processed cell is pushed onto the stack of the current thread and triggered by itself (lines 36, 27).

Once a thread has finished processing all elements in its stack, it enqueues itself into a thread queue in the thread handler (line 4). Should this thread have been the last thread to do so, new grains are thrown (line 6).

4.3 Implementation in Java

Every information mentioned before can be compacted into two bytes allowing a rather low memory usage. This information comprises a flag whether a cell has been assigned to be processed by a thread, the kind of cell position in the cache-optimized array and the cell state itself. The layout of these two bytes is illustrated below:



There are five kinds of cell positions in the cache-optimized array. Either the neighborhood is entirely inside one block (see Sect. 4.1), overlapping into a preceding or following block, entirely in a possible smaller block at the end or in this block but overlapping to the last preceding normal sized block. Depending on this classification the neighboring cells are calculated manually when firing.

Making use of the `java.util.concurrent` framework [2] efficient and fast hardware locking mechanisms (e.g. compare-and-swap) became available in Java. In the sandpile CA the regions of code which have to be executed atomically (writes to cell states) are very small and as a result, thread waiting times before such regions are short if another thread is already occupying this region. That's why spinlocks can be employed as a non-blocking locking mechanism. Every other locking mechanism would involve too much overhead for waking up and setting threads asleep. There is exactly one lock for each cell in the sandpile grid. The `lock()` method of a spinlock is simple, with `lock` being an `AtomicInteger`:

```
while (!lock.compareAndSet(0, 1)){ ; }
```

Accordingly, the `unlock()` method is as follows:

```
lock.set(0);
```

Thus, the code segment in which a cell update takes place can be seen here for the left neighboring cell. It corresponds to lines 32-37 in Algo. 3.

```

leftLock.lock();
if (((caValues[left] += nrOfIncs) & checkBits) > 3) {
    caValues[left] |= setAvailPosBit;
    leftLock.unlock();
    if ((tempThread = threadHandler.deq()) != null){
        tempThread.startCell = left;
    } else {
        cellStack.push(left);
    }
} else {
    leftLock.unlock();
}

```

In this code excerpt `nrOfIncs` indicates how many times the current triggering cell has to fire (see Sect. 4.2). All neighboring cells are then incremented atomically by this value. `checkBits` is a bit mask masking only bit no. 1 and bits 5-16 (see above). This permits easy checking for criticality as a cell would be negative with the availability bit set at position no. 1.

Another point where locking mechanisms have to be used is in the thread handler. There, the return procedure has to be synchronized as threads have to be enqueued atomically. Dequeuing takes place using one lock per queue. That distributes the load as requests for dequeuing happen a lot more often than enqueueing which can be deduced from the above code segment (inner if-statement) making a call to the dequeue method every time a critical cell is encountered. Using several queues instead of one helps avoiding a bottleneck.

Something which is also important to note is that at certain points lazy behaviour is used in order to avoid additional locking actions. This can only be done if the correctness is left intact. An example here is the dequeuing procedure from the queues ring in the thread handler as the queue to dequeue from might already have been emptied by another thread between the current thread checking if a queue is full (Algo. 3, line 33) (doesn't happen atomically in order to only approximately find out where a new thread might be dequeued) and actually dequeuing from it (Algo. 3, line 34) (happens atomically in order to prevent race conditions). If the current thread fails to dequeue a new thread the cell that would have had to be passed on is simply pushed onto its own stack. Thus, a thread does not have to make reservations at a queue which would create an unnecessarily long exclusive section and as a result slow down execution.

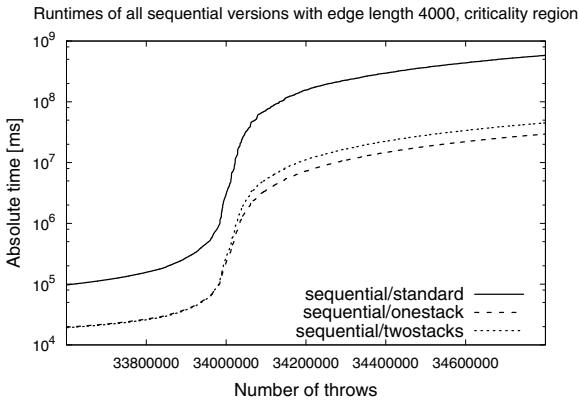
5 Evaluation

Figure 2 shows the test results on an AMD Opteron multicore computer with 16 cores using implementations of both sequential and parallel algorithms in Java.

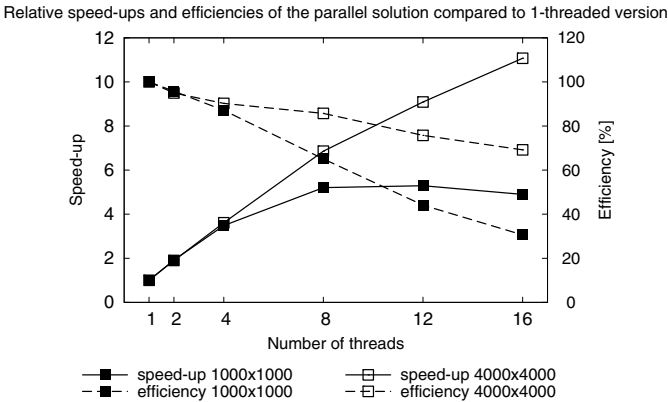
A Java implementation of the solution in 3 is referred to as “sequential standard solution”. For the sequential solutions in Fig. 2a it can be seen that the solution with one stack runs considerably faster than the synchronous solution for a grid with side length $n = 4000$. Runtimes for the other tested side lengths $n = 1000$ and $n = 2000$ can't be shown due to lack of space. For a grid size of 1000×1000 both solutions are still equally fast.

In the next Fig. 2b speed-ups and efficiencies compared to the parallel solution itself for $n = 1000$ and $n = 4000$ and 1 to 16 threads are illustrated. For $n = 4000$ a speed-up of approximately 11.5 is reached for 16 threads yielding an efficiency of around 70%.

The two most important figures are Fig. 2c and 2d. Fig. 2c shows the speed-up of the parallel solution compared to the implementation of the solution in [3] with the parallel solution being faster than this version already with only one thread. This also demonstrates that the parallel solution itself is efficient. Fig. 2d then shows the speed-up compared to the fastest sequential algorithm presented in this paper leading to speed-ups of below 2 for grids of size 2000×2000 and bigger. That is a quite bad efficiency but absolute time gains are still valuable.



(a)



(b)

Fig. 2. Evaluation results

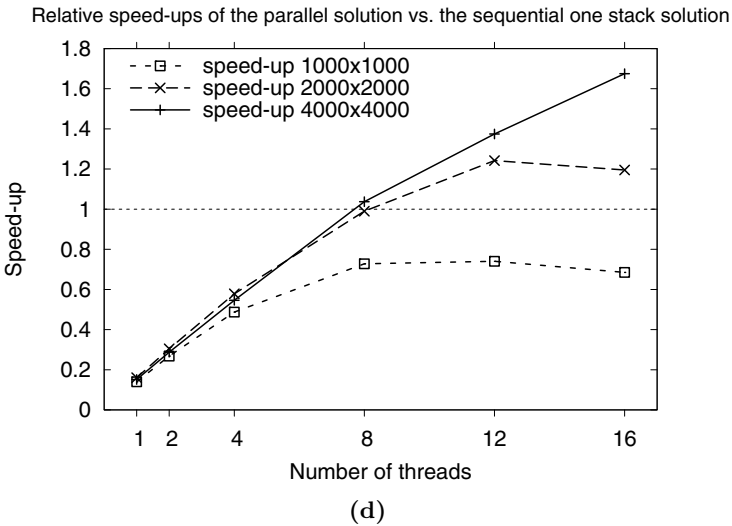
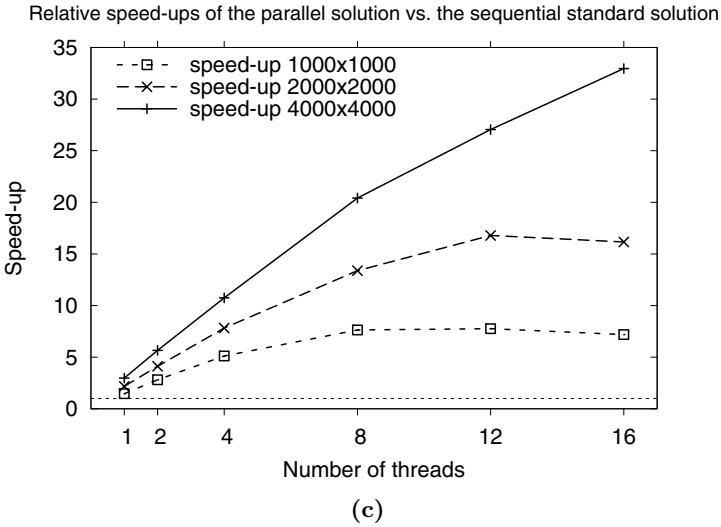


Fig. 2. (continued)

For $n = 1000$ and $n = 2000$ it is clearly visible that the best performance is available with 12 threads. Adding more threads slows down execution. That is due to too “small” avalanche sizes which is not the case for $n = 4000$.

The cache optimization technique introduced in Sect. 4.1 results in accelerations of around 9% with $n = 2000$ and 17% with $n = 4000$.

The following table summarizes the speed-ups of the three presented solutions compared to the solution by Walter and Worsch, emphasizing the fastest one.

Grid size	Sequential, 2 stacks	Sequential, 1 stack	Parallel (Threads)
1000 × 1000	10.85	10.49	7.76 (12)
2000 × 2000	12.05	13.52	16.79 (12)
4000 × 4000	12.89	19.67	32.96 (16)

Interestingly, the Java implementation of the sequential standard solution is 7.5% faster than the one in C in [3].

6 Conclusion and Outlook

The previous sections showed that it is possible to parallelize the sandpile model efficiently with good speed-ups. But even in the absence of a parallel computer calculations can still be done considerably quickly using the provided sequential solutions only.

What is still missing is a fast parallel algorithm using synchronous updating so that the number of global update steps of the sandpile CA can be found out. For this purpose a different approach is currently in development. This approach is lock free and might work for other cellular automata as well. Until then, the sequential two-stack solution will deliver those results. One could also investigate the possibility to use two stacks for the parallel stack solution instead of one. Each thread would then be calculating synchronously and possibly a way could be found to calculate the overall update steps from this information.

As longer runs are now possible in less time evaluations of these long runs have shown that the occurrences of the quiescent states in stable configurations do not remain constant when reaching self-organized criticality. It seems to be the case that occurrences of state 0 rise by the same amount as occurrences of state 2 fall. The same is true for states 3 and 1 only with a slighter change.

Not only for this interesting finding may the provided solutions offer a faster possibility for future research.

References

1. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality. *Phys. Rev. A* 38(1), 364–374 (1988)
2. Lea, D.: The `java.util.concurrent` synchronizer framework. *Sci. Comput. Program.* 58(3), 293–309 (2005)
3. Walter, R., Worsch, T.: Efficient Simulation of CA with Few Activities. In: Sloot, P.M.A., Chopard, B., Hoekstra, A.G. (eds.) *ACRI 2004*. LNCS, vol. 3305, pp. 101–110. Springer, Heidelberg (2004)

Theory and Application of Equal Length Cycle Cellular Automata (ELCCA) for Enzyme Classification

Soumyabrata Ghosh, Tirthankar Bachhar, Nirmalya S. Maiti,
Indrajit Mitra, and P. Pal Chaudhuri

Cellular Automata Research Lab (CARL), Alumnus Software Ltd. Salt Lake,
Sector - V, Kolkata 700091, India
soumyab.g@gmail.com

Abstract. A special class of n cell null boundary invertible three neighborhood CA referred to as Equal Length Cycle CA (ELCCA) is proposed in this paper to represent the features of n bit symbol strings. Necessary and sufficient conditions for generation of ELCCA has been reported. A specific set of ELCCA cycles are selected by employing the mRMR algorithm [2] popularly used for feature extraction of symbol strings. An algorithm is next developed to classify the symbol strings based on the feature set extracted. The proposed CA model has been validated for analyzing symbol string of biomolecules referred to as Enzymes. These biomolecules are classified on the basis of the catalytic reaction they participate. The symbol string classification algorithm predicts the class of any input enzyme with accuracy varying from 90.4% to 98.6%. Experimental results have been reported for 22800 enzymes with wide variation in species.

Keywords: ELCCA(Equal Length Cycle Cellular Automata), Symbol String Analysis, Enzyme Classification.

1 Introduction

A Cellular Automata (CA) model for analysis of symbol strings is presented. It employs a special class of CA referred to as Equal Length Cycle CA (ELCCA). The model has been validated for classification of symbol strings of biomolecules referred to as enzymes that control all metabolic functions of a living organism. An enzyme is synthesized out of coding sequence of DNA string represented as a sequence of four nucleotides - Adenine (A), Thymine (T), Cytosine (C), Guanine (G). A triplet of three nucleotides is referred to as codon. A codon can be encoded as a six bit symbol while assigning two bits for each of the four nucleotides. Thus an enzyme can be viewed as a symbol string, each symbol representing one of 64 codons.

Symbol string analysis is traditionally handled as pattern analysis and classification problem. A large number of authors [1-2],[14-15] has contributed in the field of pattern classification/ recognition. The model based pattern classifiers are the Nave Bayes classifier (NB) [14] and the Gaussian (Gauss) classifier [15].

Enzymes are classified based on their functionalities in metabolic pathways. Automated classification procedure of enzymes from its sequence is reported in [13],[16-17]. In [13], a machine-learning method is used to classify enzyme without using sequence similarity. The proposed method predicts the class with 74% success rate. Most of the other classification methods [20] are based on sequence similarity and applied on a small set of enzymes.

In the background of phenomenal growth of string processing applications in the field of Bio-Informatics [3], this paper reports a Cellular Automata (CA) model for analysis of symbol strings. The robustness of the model has been tested for classifying enzymes. The proposed algorithm predicts the class of an enzyme with accuracy varying from 90.4% to 98.6%. Experimental results on classifying 22800 enzymes are reported.

The field of Cellular Automata (CA) has been enriched by a large number of authors [4-7],[18-19]. The CA preliminaries is reported in Section 2 followed by ELCCA characterization in Section 3. Symbol string analysis is presented in Section 4, while Section 5 reports the classification of biomolecules known as enzymes.

2 Cellular Automata Preliminaries – Rule Vector Graph (RVG)

The Rule Vector (RV) of an n cell null boundary CA is denoted as $\langle R_0 R_1 \dots R_i \dots R_{n-1} \rangle$ where rule R_i is employed on the i^{th} cell. RVG construction from the Rule Vector (RV) of a CA has been detailed in [7], that reports (a) a linear time algorithm to identify invertibility of a CA, and (b) all the NRS (Non-Reachable States) and Self Loop States. RVG provides the foundation for the analysis of Equal Length Cycle CA (ELCCA) employed for string processing reported in the current paper. RVG construction and analysis are briefly introduced, followed by a few basic definitions used to explain RVG of a CA.

Definition 1: Rule Min Term (RMT) - The 8 Minterms of the 3 variable Boolean function f_i , corresponding to the rule R_i employed on i^{th} cell is referred to as RMTs. The three Boolean variables are a_{i-1} , a_i , a_{i+1} , the current state values of $(i-1)^{th}$, i^{th} , $(i+1)^{th}$ cells respectively, whereby the minterm $m = \langle a_{i-1} a_i a_{i+1} \rangle$. $T(m)$ denotes a single RMT in the text and it is noted simply as m for clarity of figures. The symbol $\{T\}$ represents the set of all of the 8 RMTs, whereby $\{T\} = \{T(0), T(1), T(2), T(3), T(4), T(5), T(6), T(7)\} = \{T(m)\}$. In general, a single RMT for i^{th} cell is also denoted as $T^i \in \{T\}$, where $T^i = \langle a_{i-1}, a_i, a_{i+1} \rangle$.

Definition 2 : 0-RMT and 1-RMT - For a specific CA rule R_i , the next state value b_i of i^{th} cell is '0' for a subset of RMTs while for the other subset the value is '1'. Hence a CA rule divides the RMTs into two subsets referred to as 0-RMT and 1-RMT respectively, which are denoted as $\{T_0^i\}$ and $\{T_1^i\}$, where $\{T_0^i\} \cap \{T_1^i\} = \phi$ and $\{T_0^i\} \cup \{T_1^i\} = \{T\}$.

Definition 3: A Node - The symbol $\{V\}$ representing a subset of RMTs is used to denote a node in the Rule Vector Graph (RVG). The i^{th} level RVG corresponding to the rule R_i consists of a set of input and output nodes connected by directed edges. An output node of i^{th} level is derived from its input node through RMT transition (Fig 1(b)). The output node of i^{th} level is the input node of $(i + 1)^{th}$ level corresponding to the rule R_{i+1} .

Definition 4 : Edge of a Level - It represents RMT transition from input to output node. Let $T^i = T(m) = \langle a_{i-1} a_i a_{i+1} \rangle = \langle 1 0 1 \rangle$ be a RMT of an input node. Consequently, T^{i+1} (a RMT of output node) can be derived from T^i as - $T^{i+1} = \langle a_i a_{i+1} a_{i+2} \rangle = \langle 0 1 0 \rangle = 2$ and $\langle 0 1 1 \rangle = 3$ by deleting a_{i-1} and appending 0 and 1 as a_{i+2} . Fig 1(b) shows this RMT transition for all the 8 RMTs.

Definition 5 : 0-edge/1-edge and edge weight - The 0-edge and 1-edge refer to the edges from an input node of i^{th} level corresponding 0-RMT and 1-RMT for the rule R_i employed on the i^{th} cell. The b_i edge ($b_i \in \{0,1\}$) is assigned the edge weight $\{ T_{b_i}^i \} / b_i$, $\{ T_{b_i}^i \}$ represents the set of RMTs for rule R_i for which the next state value is b_i .

Definition 6 : RMT string - The CA state can be expressed as a RMT string $\langle T^0 T^1 \dots T^i \dots T^{n-1} \rangle$ where $T^i \in \{T\} = \{T(0) T(1) T(2) T(3) T(4) T(5) T(6) T(7)\}$, and $T^i = \langle a_{i-1} a_i a_{i+1} \rangle$. Thus T^i denotes the decimal value of the bit string $\langle a_{i-1} a_i a_{i+1} \rangle$, where a_{i-1} , a_i and a_{i+1} represent the current state of the $(i - 1)^{th}$, i^{th} and $(i + 1)^{th}$ cell respectively.

Definition 7 : Compatible RMT Pair - A pair of RMTs T^i and T^{i+1} in a RMT string $\langle \dots T^{i-1} T^i T^{i+1} \dots \rangle$ (where $T^i = \langle a_{i-1} a_i a_{i+1} \rangle$ and $T^{i+1} = \langle a'_i a'_{i+1} a'_{i+2} \rangle$) are compatible if (i) $a_i = a'_i$ and (ii) $a_{i+1} = a'_{i+1}$. The pair T^i and T^{i+1} is referred to as Incompatible if these are not compatible.

Definition 8 : Valid RMT String - A RMT string $\langle T^0 \dots T^{i-1} T^i T^{i+1} \dots T^{n-1} \rangle$ representing the state of a CA is a valid RMT string if each pair T^i and T^{i+1} ($i = 0$ to $(n-2)$) is a compatible RMT pair.

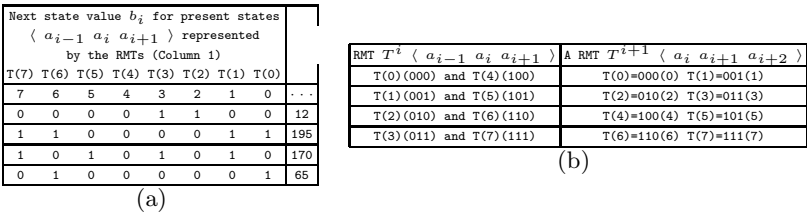


Fig. 1. (a)Rule Min Term (RMT) and CA rule (b) RMT transition (Note : The left column refers to the RMTs of i^{th} cell while the right column refers to the corresponding RMTs of $(i + 1)^{th}$ cell.)

The RVG of an n cell CA ($\langle R_0 R_1 \dots R_i \dots R_{n-1} \rangle$) is designed by concatenating i^{th} level subgraph derived for rule R_i . Fig 2 shows the RVG and STG of

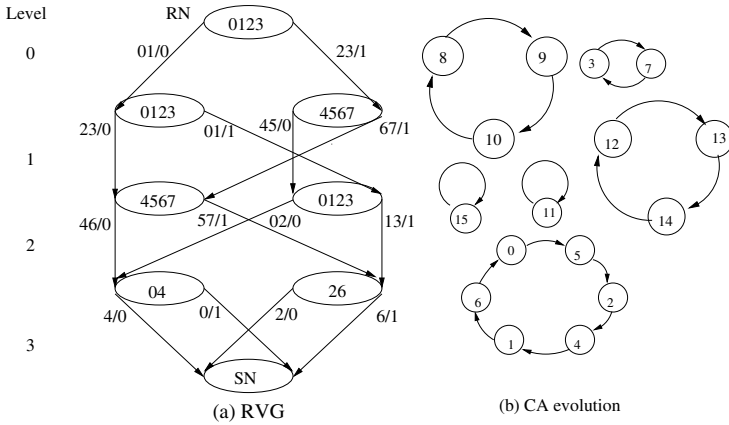


Fig. 2. RVG of the invertible CA $\langle 12\ 195\ 170\ 65 \rangle$ and its evolution

a hybrid CA $\langle 12\ 195\ 170\ 65 \rangle$. In this figure binary bit string of a CA state is denoted by its decimal value. A RMT $T(m)$ is also noted simply as m ($m = 0$ to 7) for clarity of figures and text. Because of null boundary, $\{T(0), T(1), T(2), T(3)\}$ represents the Root Node (RN) and only the even valued RMT can exist on the 3rd level ($(n - 1)^{th}$ level) input nodes. As per Rule 12 (Fig 1(a)), the level 0 edges are drawn from the RN - the 0-edge with weight $\{T(0), T(1)\}/0$ and 1-edge with weight $\{T(2), T(3)\}/1$. Level 0 output nodes $\{T(0), T(1), T(2), T(3)\}$ is derived out of RMTs $T(0), T(1)$ as per RMT Transition (Fig 1(b)). The other output node $\{T(4), T(5), T(6), T(7)\}$ is derived out of RMTs $T(2), T(3)$ on 1-edge. Level i ($i = 0, 1, 2$) output nodes act as the input nodes for level $(i + 1)$. The i^{th} level RVG for $i = 1, 2, 3$ are drawn as per rules $R_1 = 195, R_2 = 170, R_3 = 65$ (Fig 1(a)). Level 3 output node is marked as SN (Sink Node). Traversal of RVG of Fig 2(a) generates the State Transitions noted in Fig 2(b). RVG analysis (Theorem 3 in [7]) establishes that it is an invertible CA.

If all the cycles of an invertible CA are of same length, it is referred to as Equal Length Cycle CA (ELCCA). Fig 3 shows the RVG and its evolution of a 4 cell ELCCA having two cycles, each of length 8. An n cell ELCCA generating cycles of length k are referred to as ELCCA(n,k).

The main contribution of the current paper in the field of CA theory follows next.

3 Level Graph(LG) to Characterize CA Evolution

Detailed analysis of CA evolution has been formulated based on another graph referred to as Level Graph (LG) derived out of the RVG of an n cell CA. The Level Graph $LG(i)$ for i^{th} level of RVG ($i = 0, 1, \dots, (n - 1)$) of an n cell CA can be derived as follows.

Each node in a LG is a RMT, while its edge (v_1, v_2) specifies the node v_2 reachable from the node v_1 . While number of nodes for each LG is 8, for $LG(0)$

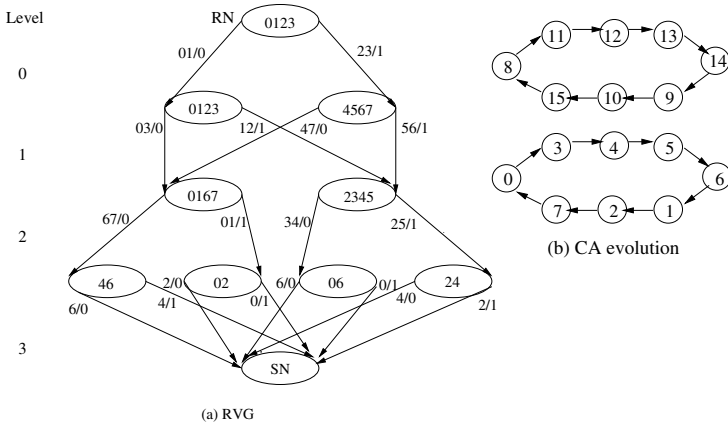


Fig. 3. An ELCCA $\langle 12\ 102\ 39\ 17 \rangle$

(0^{th} Level) and LG_{n-1} (for $(n-1)^{th}$ Level) the number of nodes is 4 due to null boundary. Generation of $LG(i)$ for level i is noted in Algorithm 1. It accepts $(i-1)^{th}$, i^{th} , and $(i+1)^{th}$ levels of RVG as input and generates the $LG(i)$. The valid path with compatible RMTs on $(i-1)^{th}$, i^{th} , and $(i+1)^{th}$ level is considered for generation of $LG(i)$, $i=0$ to $(n-1)$.

Algorithm 1: Generate $LG(i)$ (for level i of RVG)

Input: The $(i-1)^{th}$, i^{th} , and $(i+1)^{th}$ levels of RVG

Output: $LG(i)$

Step 0: Consider RMT on i^{th} level edge as a node.

Step 1: For each RMT (a node) at i^{th} level consider the input of possible next states $\langle b_{i-1}b_i b_{i+1} \rangle$ on the valid path covering edges on $(i-1)^{th}$, i^{th} , and $(i+1)^{th}$ level. Convert it to the decimal number and mark it as the terminating node in the $LG(i)$.

Step 2: Connect the two nodes with an edge.

Step 3: Repeat Step 1 and 2 for each RMT of the level i . Stop

Fig 4 illustrates the six LGs of a (6,8) ELCCA (6 cell CA with 8 cycles each of length 8). In the subsequent discussions a node of CA evolution corresponding to State Transition Graph (STG) (Fig 4(c)) refer to a CA state. On the other hand a node of LG (Fig 4(b)) refers to a RMT. A $LG(i)$ may have multiple components, each component is a subgraph covering a subset of nodes with no incoming or outgoing edge. For example, $LG(3)$ (Fig 4(b)) of the CA $\langle 5\ 240\ 165\ 15\ 204\ 65 \rangle$ has two subgraphs - Subgraph 1 covering nodes 0, 2, 4, 6 with a self loop on node 2 and 4, while other one (Subgraph 2) covers nodes 1, 3, 5, 7 with self loop on node 3 and 5. Each CA state represented as a valid RMT string $\langle T^0 T^1 \dots T^i \dots T^{(n-1)} \rangle$ has its RMT T^i as a node of $LG(i)$. The 0^{th} cycle of Fig 4(c) covers CA states (denoted as - Decimal Value (Binary) \langle RMT string \rangle) - 0(000000) \langle 000000 \rangle , 45(101101) \langle 253652 \rangle , 48(110000) \langle 364000 \rangle , 21(010101) \langle 125252 \rangle , 12(001100) \langle 013640 \rangle , 33(100001) \langle 240012 \rangle , 60(111100) \langle 377640 \rangle , 25(011001) \langle 136412 \rangle

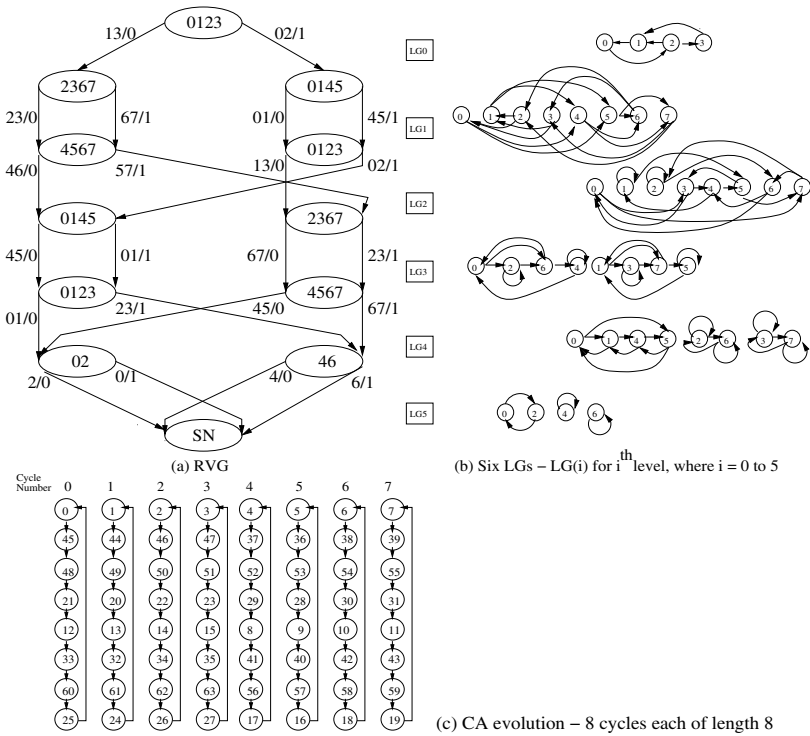


Fig. 4. An example(6,8) ELCCA - 6 cell CA $\langle 5\ 240\ 165\ 15\ 204\ 65 \rangle$ having 8 cycles each of length 8 - its RVG, six LGs, and eight cycles

with RMT T^3 as 0, 6, 0, 2, 6, 0, 6, 4 (underlined in RMT string) for eight CA states. Subgraph 1 of $LG(3)$ covering nodes 0,2,4,6 is associated with this cycle (marked as Cycle Number 0 in Fig 4(c)). The Subgraph 1 is also associated with 3 more cycles marked as Cycle Number 1, 4, 5. On the other hand Subgraph 2 of $LG(3)$ (Covering states 1,3,5,7) generates Cycle No. 2 with states 2 ($\langle 000124 \rangle$), 46 ($\langle 253764 \rangle$), 50 ($\langle 364124 \rangle$), 22 ($\langle 125364 \rangle$), 14 ($\langle 013764 \rangle$), 34 ($\langle 240124 \rangle$), 62 ($\langle 377764 \rangle$), 26 ($\langle 252524 \rangle$) in addition to 3 other cycles with Cycle No. 3, 6, 7.

Fig 5 illustrates the RVG and evolution of ELCCA(6,16) - 6 cell CA $\langle 6\ 125\ 240\ 86\ 80 \rangle$ with 4 cycles each of length 16. The $LG1$ (of level 1 RVG) has two subgraphs, each having 4 nodes - subgraph 1 with nodes 0, 1, 2, 7, while subgraph 2 covers nodes 3, 4, 5, 6. The following lemma and theorem characterize ELCCA evolution.

Lemma 1: An n cell CA generates a self loop CA state if each $LG(i)$ of its RVG has a self loop node/RMT with compatible RMTs for $i = 0$ to $(n - 1)$.

Proof: Let a state with valid RMT string (Defn 8) be denoted as $\langle T^0\ T^1\ \dots\ T^i\ \dots\ T^{(n-1)} \rangle$. If each T^i forms a self loop state in the $LG(i)$ $i = 0$ to $(n-1)$, then the corresponding CA state is a Self Loop state.

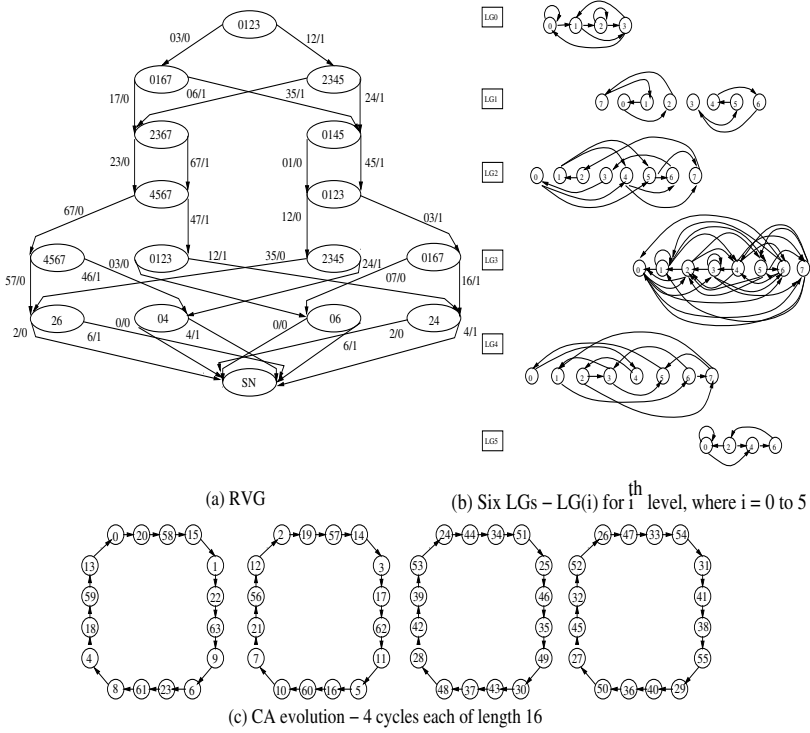


Fig. 5. An example(6,8) ELCCA - 6 cell CA $\langle 6 \ 125 \ 240 \ 86 \ 80 \rangle$ having 8 cycles each of length 16 - its RVG, six LGs, and eight cycles

Theorem 1: At least one $LG(i)$ ($i = 0, 1, \dots, (n - 2)$) of the RVG of an n cell ELCCA has even number of isomorphic sub-graphs with a valid RMT strings $\langle T^0 T^1 \dots T^i \dots T^{n-1} \rangle$ generating the cycles of the CA.

Proof: Necessity - An ELCCA has cycles of equal length. The CA states covered by each of these cycles is generated out of the nodes of LGs. Hence it is necessary that at least one LG has even number of isomorphic subgraphs in order to generate cycles of an ELCCA.

On the other hand if isomorphic subgraphs are available in a LG (say $LG(i)$), then the compatible nodes (RMTs) in $(i-1)$, $(i-2)$, .. 0^{th} LG and $(i+1)$, $(i+2)$, .. $(n-1)^{th}$ LG, generate CA states $\langle T^0 T^1 \dots T^i \dots T^{n-1} \rangle$ forming a cycle in CA STG. Because of isomorphic subgraphs, even number of equal length cycles get generated since the total number of state 2^n is even. Hence the specified condition is sufficient to generate an ELCCA.

The $LG(3)$ of the example ELCCA of Fig 4(b) has two isomorphic subgraphs out of which 8 cycles (Fig 4(c)) get generated. There are other interesting results of ELCCA characterization, that can be derived out of the RVG and its LGs of an n cell CA for any value of n . Large number of (n, k) ELCCA exists even for small value of n .

4 Equal Length Cycle CA (ELCCA) for Symbol String Classification

The problem addressed in the rest of the paper deals with analysis of symbol strings leading to their classification based on the specified features of the strings.

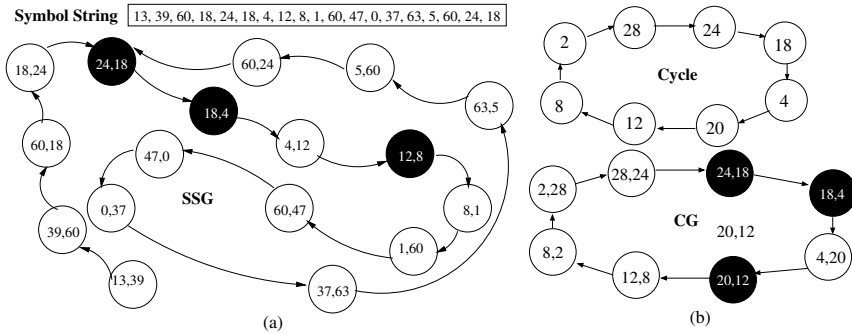


Fig. 6. (a) a 6 bit symbol string with symbols expressed in decimal value and its SSG; (b) an ELCCA(6,8) of cycle length 8 and its CG

4.1 ELCCA Modeling Feature Set of Symbol String

The feature set of the given set of n bit symbol strings are represented with ELCCA(n, k) – n cell ELCCA having k length cycles. An n bit state in n cell ELCCA cycle represents an n bit symbol of the string, while the cycles model its features. A simple graph theoretic framework has been developed to implement the Sensing Mechanism [3].

Symbol String Graph(SSG) and Cycle Graph(CG) - An n bit symbol string is represented with a graph referred to as Symbol String Graph (SSG), while Cycle Graph (CG) represents a cycle of ELCCA(n, k). The SSG (Fig 6(a)) of an example 6 bit symbol string is generated with 17 unique nodes, each, representing a pair of consecutive symbols. A pair of nodes having a common symbol is connected by directed edge. Similarly, the CG for an ELCCA(n, k) cycle (Fig 6(b)) with n bit symbols is generated by considering the cycle as a string of n bit symbols. The selection of ELCCA cycles is based on the coverage of SSG nodes by CG nodes with a parameter referred to as Global Coverage Index (GCI).

Global Coverage index(GCI) estimation - The procedure for embedding CG in SSG is discussed with reference to the CG of Fig 6(b) in the SSG of Fig 6(a). There are three nodes (marked dark) in the SSG which are common in CG. The nodes are marked in black in the figure. The path length is specified by the number of nodes traversed in the paths. In the SSG Fig 6(a), there are two

paths, $(24,18) \rightarrow (18,4)$ of length 2 and one degenerate path with single node $(12,8)$ of length 1. The GCI value is computed as follows.

$$GCI = \sum_i N_i^2, \quad N_i = \text{length of } i^{\text{th}} \text{ path}$$

The GCI value on embedding CG in SSG (Fig 6) is given by -

$$GCI = 2^2 + 1^2 = 5$$

The estimated GCI value is employed in the next section to select the ELCCA(n,k) cycle set used for analysis and classification of n bit symbol strings.

4.2 Classification of Symbol Strings

The ELCCA based classification scheme, as proposed in [3], implements the following three phases.

(I) Sensing : In the sensing mechanism, each of the training symbol strings from m number of classes are analyzed with ELCCA(n, k) cycles to generate the parametric values GCI (Global Coverage Index).

(II) Feature Extraction : To extract features for m number of classes, ${}^m C_2$ number of pair-wise classifiers are used. The distinguishing features are extracted from a pair of classes at a time. Subsequently, the mRMR algorithm [2] extracts a feature set which can differentiate the training symbol strings into two classes on the basis of estimated GCI. For each of the pair-wise combinations of Class A and B, the selected features are classified into two-groups : (i) $F_{present}^A$ - The features which are presented in Class A and absent in Class B; and (ii) F_{absent}^A - The features which are present in Class B and absent in Class A. A feature set is represented by a collection of ELCCA(n, k) cycles. The GCI values estimated from CG of these ELCCA(n, k) cycles and SSG of the training symbol strings should ensure significant difference between Class A and B types of strings.

(III) Classification based on Probabilistic Decision Making System : On the basis of the extracted feature set, a probabilistic decision making system is developed to predict the class of an input testing symbol string. From a pair-wise combination, a probability of the input symbol string being in each class can be calculated from its GCI value with respect to the selected feature set. This procedure is executed for ${}^m C_2$ number of pair-wise combinations. The probability values of the input testing symbol string being in each of the m classes are computed by Compound Probability. This method predicts the class of the testing symbol string as the one for which the computed compound probability is maximum.

5 Enzyme Classification with 6 Cell ELCCA

Robustness of the CA model for classification of symbol string is tested for a real life application that deals with classification of enzymes. Enzymes are protein molecules those control metabolic functions in living organisms. Classification of enzymes on the basis of the nature of chemical reaction they catalyze and

their substrate specificity is an extensively researched topic in Biology [12,13]. Enzymes can be viewed as symbol strings, where each symbol is a codon represented with 6 bits. The 6 cell ELCCA cycles are employed for analysis and classification of 6 bit symbol strings of enzymes. All the available 97812 number of ELLCA(6,8) are considered for this classification procedure.

The training data set to build the ELCCA model is collected from Exspasy Swis-port Enzyme Nomenclature Database [8] where all enzymes are classified according to Enzyme Commission(EC) Classification System [10]. This is a four level (Fig 7) enzyme classification system based on chemical reaction and substrate specificity. The general structure of the class nomenclature is $E_1.E_2.E_3.E_4$, where E_1 specifies the highest level class and E_4 is the lowest one. The lowest level of enzyme classification mainly represents the substrate specificity of same enzymatic reactivity. For example, the difference between class 1.1.1.1 and 1.1.1.3 is the substrate Alcohol or Homoserine. Other reaction properties of both the classes 1.1.1.1 and 1.1.1.3 are identical. For our current study, only Third Level (e.g. 1.1.1.*) Classes are classified into its sub-classes (e.g. 1.1.1.1, 1.1.1.2 etc). The ELCCA model classifies enzymes with same chemical reactivity into sub-classes having different substrate specificity.

Enzyme Classes	Name	Level
1 . . . *	Oxidoreductases.	1st Level Class
1 . 1 . * *	Acting on the CH-OH group of donors	2nd Level Class
1 . 1 . 1 . *	With NAD(+) or NADP(+) as acceptor	3rd Level Class
1 . 1 . 1 . 1	Alcohol dehydrogenase	4th Level Class
1 . 1 . 1 . 2	Alcohol dehydrogenase (<i>NADP</i> (+)).	4th Level Class

Fig. 7. Enzyme Commission(EC) Classification System for Homoserine Dehydrogenase with $E_1 = 1$, $E_2 = 1$, $E_3 = 1$, $E_4 = 1, 2, 3, - - -$

5.1 Experimental Procedure

The three phases of symbol string classification, as noted in Section 4, are implemented for enzyme classification based on the analysis of Coding DNA Sequence (CDS) of the respective enzyme. These CDS are collected from Uniprot Protein Knowledge base [9] related to Exspasy Swis-port Enzyme Nomenclature Database [10]. CDS sequences highly variant in respect of average length of the members in that class are filtered out. The filtered dataset is organized in four-level classes according to the Exspasy Enzyme Classification. Next, the dataset is separated into two-third and one-third population for training and testing purpose respectively. For classes containing large number of entries, randomized sample selection procedure is used to select training and testing set based on K-fold Iteration, where $K = 10$ [11]. For the complete experiment, we have trained and tested the model with 38 Third Level Classes. The ELCCA based classifier is designed for each of the Third Level Classes which can classify an input CDS sequence of the candidate enzyme into its sub-classes tabulated in Fig 7. The optimal size(p) of number of features in the feature set has been identified as 50 derived out of 97182 number of ELCCAs(6,8).

5.2 Results and Discussions

Results of 38 Multi-class classifiers are presented in Fig 8. Total number of enzymes tested is more than 22800 with variation in species. The accuracy of classification varies from 90.4% to 98.8%. Errors are mostly from false positive prediction with a negligible number of false negative cases. There is only one case of 84.2 % correct prediction for the (* marked) enzyme class 1.2.2 due to non-inclusion of proper ELCCA(6,8) cycles.

Enzyme Classes	Number of Sub-classes	False Positive (%)	False Negative (%)	Success Rate (%)	Enzyme Classes	Sub-classes	False Positive (%)	False Negative (%)	Success Rate (%)
1.1.2	28	2.1	0	97.8	2.2.1	21	1.8	2.3	95.8
1.1.3	26	1.2	0	98.6	2.3.1	18	1.4	0	97.8
1.2.1	21	3	0	96.8	2.3.2	6	1.7	0	98.1
* 1.2.2	23	14.8	1.1	84.2	2.3.3	13	1.7	0	97.6
1.2.3	31	2.8	0	97.1	2.7.1	7	2.6	0	97.2
1.2.1	22	1.8	0	98.0	2.7.2	8	1.4	2.1	96.5
1.3.1	28	3.5	0	96.2	2.1.3	8	3	2.1	94.9
1.3.2	12	6.4	3.2	90.4	3.1.1	29	2.7	0	97.1
1.3.4	6	2	0	97.6	3.1.2	14	2.8	0	96.8
1.3.6	6	4.5	0	94.9	3.2.1	21	2.6	2.3	95.1
1.4.1	18	6.5	0	92.8	3.2.2	11	1.7	0	98.1
1.4.3	8	4.3	0	95.4	3.4.1	28	3.2	4.1	92.7
1.4.4	6	5.5	0	94.1	3.4.2	12	1.9	0	97.8
1.5.1	12	1.5	0	98.4	4.1.1	14	1.7	0	98.2
1.5.1	6	1.7	1.1	97.1	4.1.1	8	1.8	2.1	96.1
1.6.1	7	1.8	0	97.9	5.1.1	11	2.3	0	97.4
2.1.1	25	1.7	1.8	96.5	5.1.2	7	0	0	98.9
2.1.2	12	1.8	0	98.1	6.1.1	6	2.1	0	97.1
2.1.3	6	1.2	0.9	97.8	6.2.1	8	3.1	0.6	96.2

Fig. 8. Results of 38 Third Level Enzyme Classifiers indicating the percentage of False Positive and False Negative values, and Success Rate for each of the classes

6 Conclusion

A special class of CA referred to as Equal Length Cycle(ELCCA) is employed to model classification of a symbol string based on its feature set. Robustness of the scheme has been validated for classification of the biomolecules known as enzymes.

References

1. Webb, A.R.: Statistical pattern recognition, 2nd edn. John Wiley, New York (2002)
2. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(8), 1226–1238 (2005)
3. Liew, A.W.-C., Yan, H., Yang, M.: Pattern recognition techniques for the emerging field of bioinformatics: A review *Pattern Recognition* 38, 2055–2073 (2005)

4. Toffoli, T., Margolus, N.: Cellular Automata Machines A New Environment for Modeling. MIT Press, Cambridge (1987), ISBN: 0-262-20060-0
5. Chaudhuri, P.P., Chowdhury, D.R., et al.: Additive Cellular Automata: Theory and Applications. John Wiley and Sons, Chichester (1997), ISBN: 0818677171, 9780818677175
6. Wolfram, S.: New Kind of Science. Wolfram Media Inc., Champaign ISBN- 1-57955-008-8
7. Maiti, N.S., Ghosh, S., Munshi, S., Pal Chaudhuri, P.: Linear Time Algorithm to Identify Invertibility of Null-Boundary Three Neighborhood Cellular Automata. To be published on Complex Systems 19(1) (2010)
8. Expaty Swiss-Prot Enzyme nomenclature database, <http://www.expasy.ch/enzyme/>
9. UniProtKB Protein Knowledgebase by EBI, PIR and SIB, <http://www.uniprot.org/uniprot/>
10. Webb, E.C.: Enzyme nomenclature 1984: recommendations of the Nomenclature Committee of the International Union of Biochemistry on the nomenclature and classification of enzyme-catalysed reactions. In: International Union of Biochemistry. Academic Press, London (1984)
11. Davison, A.C., Hinkley, D.V., Canty, A.J.: Bootstrap Methods and Their Application, 2nd edn. Cambridge University Press, Cambridge (1999), ISBN: 9780521574716
12. Harper, H.A., Martin, D.W.: Harper's Review of Biochemistry Lange Medical Publications (1985), ISBN: 0870410385, 9780870410383
13. des Jardins, M., Karp, P.D., Krummenacker, M., Lee, T.J., Ouzounis, C.A.: Prediction of enzyme classification from protein sequence without the use of sequence similarity. In: Proc. Int. Conf. Intell. Syst. Mol. Biol., vol. 5, pp. 92–99 (1997)
14. John, G.H., Langley, P.: Estimating Continuous Distributions in Bayesian Classifier. In: Eleventh Conference on Uncertainty in Artificial Intelligence, San Mateo, pp. 338–345 (1995)
15. van der Walt, C.M.: Maximum Likelihood Gaussian Classifier (2007), <http://www.patternrecognition.co.za>
16. King, R.D., Karwath, A., Clare, A., Dehaspe, L.: The utility of different representations of protein sequence for predicting functional class. Bioinformatics 17(5) (2001)
17. Henrissat, B.: A classification of glycosyl hydrolases based on amino acid sequence similarities. Biochem. J. 280(Pt. 2), 309–316 (1991)
18. Schiff, J.L.: Cellular Automata A Discrete View of the World. Wiley & Sons, Inc., Chichester, ISBN: 047016879X
19. Chopard, B., Droz, M.: Cellular Automata Modeling of Physical Systems. Cambridge University Press, Cambridge (1998), ISBN: 0-521-46168-5

Cellular Automata Model for Size Segregation of Particles

Toshihiko Komatsuzaki and Yoshio Iwata

Institute of Science and Engineering, Kanazawa University, Kakuma-machi,
Kanazawa, Ishikawa, 920-1192 Japan
{toshi,iwata}@t.kanazawa-u.ac.jp

Abstract. This paper deals with the study of size segregation of particles where the size difference causes characteristic movement of particles inside granular media according to the induced vibration. In this study, segregation of particles due to the difference in size is simulated using Cellular Automata. A connected lattice automaton is introduced in the model, so that the variation of particle sizes as well as geometrical arrangement between particles can be represented. The Cellular Automata model can produce various characteristics which are observed in the actual granular systems. Furthermore, it is known from both numerical and experimental observations that the segregation progress is dependent on the amplitude of excitation as well as the particle size ratio.

1 Introduction

A mixture of particles whose diameter or the density are different, when filled in a container and exposed to a certain excitation, may exhibit segregation patterns where particles are eccentrically stabilized at positions according to each physical property. Among several types the size segregation is well known, where the smaller particles pass through voids formed between larger particles and eventually the latter rises upward to a free surface. Vibratory excitation increases the unfilled space, which then reduces friction between particles and accelerates their mobility. Consequently, particles with different properties are inhomogeneously located in space and form various types of stable segregation pattern under specific fluidization process. However, such pattern formation may cause adverse effect on product quality due to the non-uniformity in the course of mixture process. Segregation as a whole is thought to be undesirable phenomena in industries and a number of studies have been conducted to understand the mechanisms and to control processes.

Hayashi et. al. [1] and Ikeda et. al. [2] investigated experimentally the vibration induced segregation process of granular systems by testing various conditions, e. g., the way to supply granular materials and the amount of each size grain. They have shown that how the ratio of particle diameter affected the resultant segregation or the mixture state, and also revealed that the friction between container wall and particles causes particle motion be different in the vicinity

of wall and in the midsection of the container, which then induces upward flow in the midsection and adverse flow in the peripheral region. In such a state, the peripheral smaller particles penetrate into the central relaxed region and consequently the segregation occurs. Knight[?] also pointed out that the convection flow intermediated by the friction of the wall is a significant factor for the issue, through similar experimental investigations.

Numerical analyses are also performed in order to understand the granular related events both from the analytical and phenomenological point of view[4]-[11]. Rather than the continuum treatment of granular systems, the discrete element method (DEM) is widely used in various studies focusing on each particle behavior for understanding the gross phenomena. DEM is built on the basis of Newtonian dynamics, where the constitutive granular system behavior is built up by respective particle motion which is modeled by a set of equation of motion. Studies include vibratory conveyance system of grains[4], Modeling of fluidized beds[5], inner convective flow dynamics[6], Vibration induced segregation model[7], etc. Taguchi[6] modeled the dynamics of vibrated bed consisting of homogenous spherical particles using DEM and simulated the typical characteristics in granular systems such as convection and heaping at the central portion of the media. He pointed out that convection is sensitive to the numerical friction coefficient setting so that the flow directions become consistent with actual observations. It should also be addressed that the computational load becomes indispensable as the number of governing equations as well as the number of particles increases. However, the time step should be kept moderately small in order to prevent the numerical instability.

Challenges have been made for modeling granular related problems using Cellular Automata and are successful by defining relatively simple rules [8]-[11]. Sakaguchi et. al.[8] introduced a lattice-connected automaton model where the single particle is associated with multiple lattice nodes, which thus lead to the alleviation of lattice constraint and the variation in movement, the particle size, the interaction between particles can be enhanced. They have also simulated the oscillating granular bed and succeeded capturing typical aspects in granular systems as mentioned above, however, the particle behavior is not explicitly associated with collision and friction properties.

In this paper, the Cellular Automata model is developed for simulating the size segregation of granular system. The model consists of a mixture of two different size particles filled in a container, where the entire system is shaken by continuous tapping excitations. Based on the lattice-connected automaton model, the local interaction rule is defined physically. Experiments are also performed for the system equivalent to the model, where the effect of the excitation amplitude and frequency to the segregation time is investigated. The Cellular Automata model can produce various characteristics which are observed in the actual granular systems. Furthermore, it is known from both numerical and experimental observations that the segregation progress is dependent on the amplitude of excitation as well as the particle size ratio.

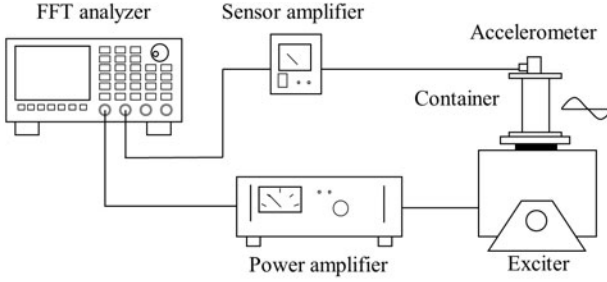


Fig. 1. Experimental setup of size-segregation of grains

2 Experiment

To capture the segregation dynamics and also to verify the Cellular Automata model as will be mentioned in the latter section, experiments are performed for granular materials consisting of two particles which are different in diameter. In this section, the overview of the experimental setup and the results are shown.

2.1 Experimental Setup

Size segregation is examined by an experimental setup shown in Fig. 1. The system consists of vertically reciprocating exciter on which a cylindrical plastic container is attached, the data analyzer, acceleration pickup, and the amplifiers. The cylinder (inner diam.: 50 [mm], height: 150 [mm]) incorporates two types of granular material made of glass beads. Two diameters are chosen out of five types, 0.6, 0.8, 1.0, 5.0 and 6.0 [mm], and five types of mixture are tested, as shown in table 1. Prior to the shaking tests, the larger particles are arranged in a layer at the bottom of the container before proceeding to pack the rest 80 [g] of smaller particles. Since the preliminary test has shown that the segregation is vulnerable to the moisture, grains are kept 24 hours inside a container with constant humidity of 80 %. Throughout experiments, the acceleration amplitude is kept 50 [m/s^2] and the plastic container is harmonically excited by the shaker within the frequency range of 20 to 200 [Hz]. In each frequency, the time required for the first larger particle to appear in a free surface is measured.

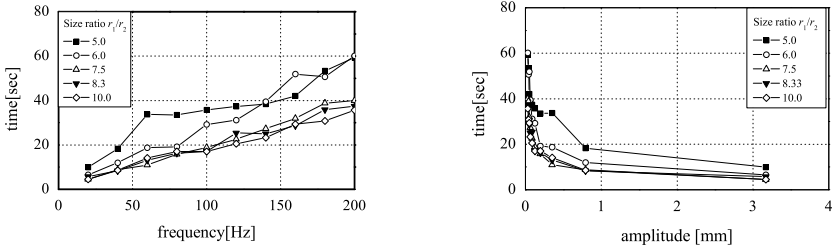
Table 1. Combination of particles

Diameter(larger)	Diameter(smaller)	Size ratio
5mm	1mm	5.0
6mm	1mm	6.0
6mm	0.8mm	7.5
5mm	0.6mm	8.33
6mm	0.6mm	10.0

2.2 Results

The measurement result of segregation time is shown in Fig. 2. In Fig. 2(a), the segregation time is plotted against excitation frequency. The segregation time is found to become large as the excitation frequency increases. It is also seen that the time become smaller with the increase of the particle size ratio. The expression in Fig. 2(a) is converted into the time against vibration amplitude as shown in Fig. 2(b), which signifies that the time become shorter with increase in excitation amplitude. It is known from these results that the excitation amplitude rather than frequency largely affects the vibration-induced segregation process.

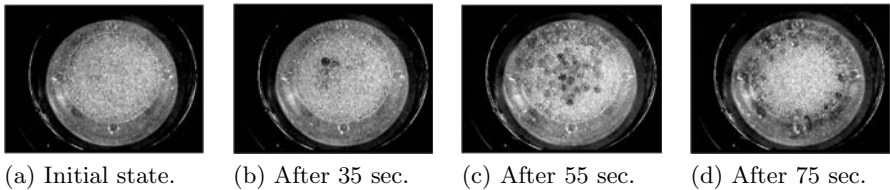
In the course of excitation, photos are taken from above the container as shown in Fig. 3, for the case the size ratio of 5 and the excitation frequency of 100 [Hz]. The larger particles segregate at the free surface in the middle of the container and then move toward the peripheral wall. They remain in the vicinity of wall without sinking into the media. The upwelling of the larger particles at center occurs along with the smaller particles, however, the smaller particles plunge near the wall according to the convection flow. Fig. 4 explains schematically the segregation process mentioned above. These features are largely supported by the past studies [1]-[3] and are commonly observed for other combinations of diameters and excitations.



(a) Segregation time v.s. frequency.

(b) Segregation time v.s. amplitude.

Fig. 2. Segregation time observed in experiment



(a) Initial state.

(b) After 35 sec.

(c) After 55 sec.

(d) After 75 sec.

Fig. 3. Segregation process of larger particles

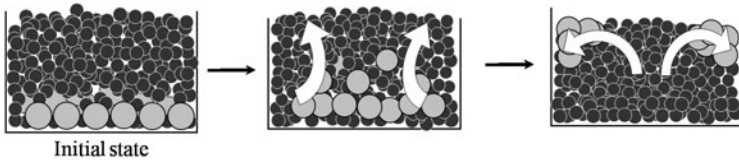


Fig. 4. Schematic of segregation process

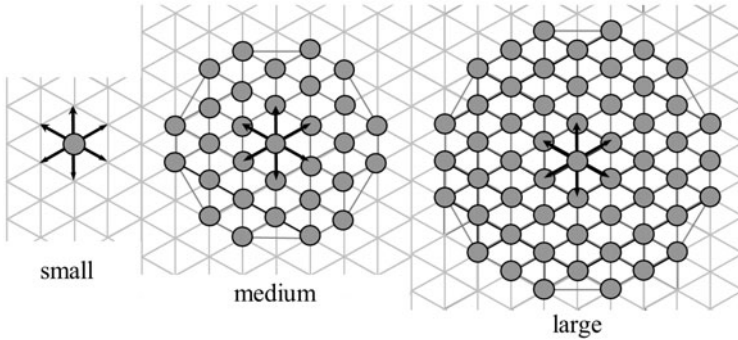


Fig. 5. Space segmentation and particle definitio

3 Cellular Automata Model of Segregation of Particles

3.1 Development of a CA Model

CA model is developed for a two-dimensional space consisting of triangular lattices. State of each lattice site is defined by three types, a segment of wall, particle and void. Additionally, apparent velocity varying within six directions is defined for particle lattices. In the present CA model, three different size particles are constituted as shown in Fig. 5. The smallest particle corresponds to single lattice, whereas the medium and the largest particles consist of 31 and 55 connected lattices, respectively. It is therefore possible to express the particle size variation and voids between particles [8](#). Note that the particles are supposed to be spherical in these expressions.

3.2 Description of the Local Neighbor Rules

Three different sets of local interaction rules representing the temporal evolution are defined for each size particle. Each set consists of particle movement, collision between particles, and also collision between particle and wall.

i) Rule set for the small particle: Particle should have the vector and move toward one of the lattice direction including stationary state, as shown in Fig. 6. If the multiple particles simultaneously move into the same site, one of them is selected by equal probability with physically consistent velocity. In the case

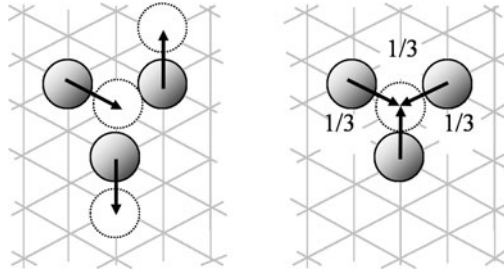


Fig. 6. Examples of rule for small particle movement, without and with conflict

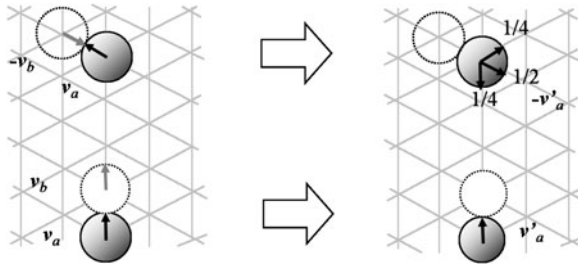


Fig. 7. Examples of rule for small particle collision

of three-body collision where the three particles are moving into the same site, equal probabilities are given for respective particles. The particles basically keep the present moving direction without collisions, however, the upward-moving particles change their moving directions to downward according to gravity if their velocity reach zero. In the present simulation model, collisions are limited to two-body cases and are evaluated between a focused and its adjacent particles. The velocity of the focused particle is changed for the colliding case. Fig. 7 shows an example of collision between small particles. In order to alleviate the lattice orientation on the particle movement after collision, the particle rebounds to the opposite direction with $1/2$ probability, and $1/4$ for the either oblique directions.

ii) Rule set for the medium particles: Since a particle is expressed by the multiple connected lattices, the variation of relative positions and the examination of collisions become more complex than the smaller particle case, however, the basic ideas for the change in velocity and the treatment of collisions are roughly the same. A couple of examples for the movement and collision of medium particles are shown in Figs. 8 and 9. The particle can move only for the case if the multiple adjacent lattices are simultaneously vacant. In the case of collision where the expected rebound direction does not coincide with the lattice orientation as shown in Fig. 9, two adjacent directions are chosen with equal probabilities, $1/2$. The change of the particle velocity after collision is determined according to the mass ratio.

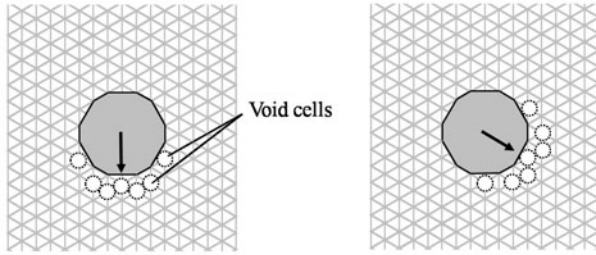


Fig. 8. Examples of rule for medium particle movement, collision between particles

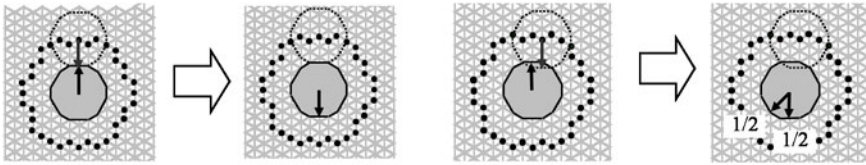


Fig. 9. Examples of rule for medium particle collision

iii) Rule set for the large particles: The rules for the large particles are basically identical to those for the medium particles, although the constitution of the reference neighbor on examining movement and collision with other is different.

3.3 Expression for the Container Excitation in Discrete System

Although the difficulty of representing continuously-changing excitation amplitude in discrete system, it is already known from the experimental observation that the segregation of particles is largely dependent on excitation amplitude rather than the excitation frequency. The vibration of the container is therefore replaced by the lift of entire granular materials at arbitrary height and the particles are let dropped freely according to gravity. Repeating this cycle apparently corresponds to steady excitation.

4 Simulation Results

Preliminary tests are first performed in order to verify the present lattice-connected Cellular Automata model for particle expression, which are followed by two cases of simulation where the particle combination is varied. Segregations are demonstrated for the granular materials consisting of small and the medium particles, and also of small and the large particles. Before starting to excite the granular media, the particles are poured randomly from above the container and are deposited inside, left over time until the count of falling particle becomes the predetermined number and the media reaches the equilibrium state. The lift of the granular media is varied from 1 to 5 grids imitating the container excitation,

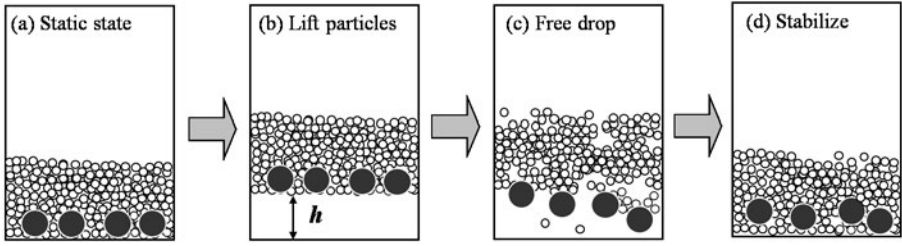


Fig. 10. Modeling of excitation process

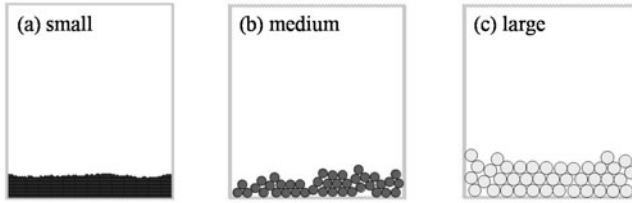


Fig. 11. Accumulation of homogeneous grains, (a) small, (b) medium, (c) large

and the results are compared for different excitation amplitude by measuring the simulation cycle until the segregation ends.

4.1 The Pile of the Homogenous Particles

Simulation results are shown in Fig.11 for respective particles with homogeneous diameter. The size of the container is given as 100 grids in both height and width. Totally 5000 particles are randomly dispersed from the opening of the container at every time step in the case of smaller particles, whereas 100 in total are dispersed for the medium and the large particles. Since the size of smaller particle corresponds to single grid, it is seen that the overall granular media is closely packed, on the other hand, in the case of medium and large particles the media is partly packed and also voids between particles are appeared. The Cellular Automata model well represents the basic profiles that are seen in the actual system related to granular materials.

4.2 Segregation of Medium Particles in Media Consisting of Smaller Particles

The segregation of medium particles submerged in a media consisting of smaller particles is simulated. Preliminary test is first conducted to verify the alternative expression of the container excitation described in Fig. 10. The system incorporates 2500 of small and 40 of medium particles, and the container is excited by lifting the gross media at $h = 4$ grids per cycle. From the results shown in Fig. 12,

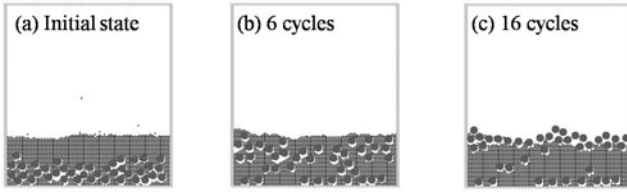


Fig. 12. Simulation result of small-medium particle segregation (2500 small, 40 medium particles)

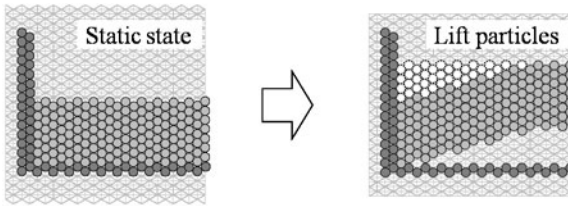


Fig. 13. The effect of friction on wall cells when lifting

the medium particles are divided into two groups, the one that segregates to the free surface, and the other remains at the bottom. The former is explained by the consequence of the insistent submerging of the smaller particles, and the latter instead is thought to be the failure of submerging due to a certain equilibrium condition between the mid and the smaller particles on lifting process. These results conform to the investigations reported by Sakaguchi, et. al.[\[8\]](#), however, as observed in the experiments all the larger particles segregated to the surface. Such a difference may arise from the friction between particles and the container wall, therefore, the effect is introduced in a simple manner such that the lift of the granular media is varied across the width as shown in Fig. 13 based on the experimental observation and the discussion in reference[\[1\]](#), that the particles in the central portion of container are less affected by the internal friction and are easy to move, whereas not for sediment adjacent to wall.

In the following simulation, the effect of the wall friction is introduced and the calculation is performed for 200 cycles. The size of the container is changed to 60 grids in height and 55 grids in width, and also the numbers of small and medium particles are 800 and 4, respectively. The four medium particles are initially located at the bottom and the maximum lift of the media is set to $h = 3$ grids at the center of the container. By 40 cycles of calculation, all of the four medium size particles emerged to the surface as shown in Fig. 14. Additionally, as observed in experiments a pile consisting of small particles is formed in the central part of the media and the segregated medium particles subsequently move toward the wall. The similar results are obtained for the combination of small and large particles as shown in Fig. 15, yet the segregation time is shortened for the larger diameter ratio. The segregation time is summarized and plotted

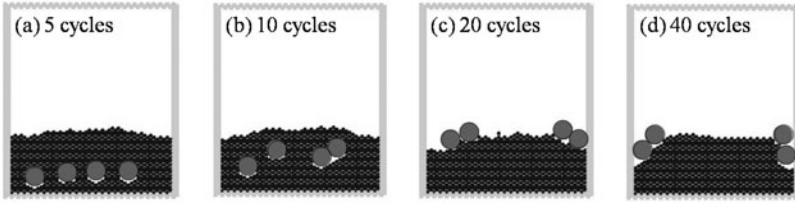


Fig. 14. Simulation result of small-medium particle segregation

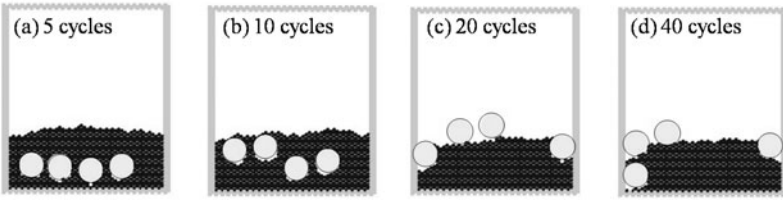


Fig. 15. Simulation result of small-large particle segregation

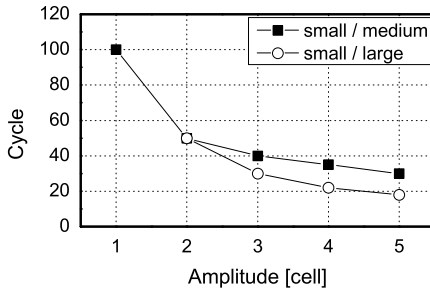


Fig. 16. Segregation time obtained by simulation

against excitation amplitude through all investigations as denoted in Fig. 16. The amount of time required for the completion of segregation becomes shorter as the excitation amplitude and the size ratio of particle increase.

5 Conclusions

In the present study, the segregation of particles where the difference in size causes characteristic movement of particles inside the granular media by the induced vibration is modeled and simulated using Cellular Automata, and the results are also compared with experimental investigations. A connected lattice automaton is introduced in the model, so that the variation of particle sizes as well as geometrical arrangement between particles can be represented. The

present Cellular Automata model can produce various characteristics such as segregation, the heaping and convective pattern of particles which are observed in the actual granular systems. Furthermore, it is known from both numerical and experimental observations that the segregation progress is dependent on the amplitude of excitation as well as the particle size ratio.

References

1. Hayashi, T., Sasano, M., Tsutsumi, Y., Kawakita, K., Ikeda, C.: Segregation in the Flow of Powder-Particles in Tapping. *Journal of the Society of Materials Science* 19(201), 574–578 (1970)
2. Ikeda, C., Tsutsumi, Y., Kawakita, K.: Segregation of Granules by Vibration. *Journal of the Society of Materials Science* 19(201), 579–582 (1970)
3. Knight, J.B., Jaeger, H.M., Nagel, S.R.: Vibration-Induced Size Separation in Granular media: The Convection Connection. *Physical Review letters* 70(24), 3728–3731 (1993)
4. Saeki, M., Minagawa, T., Takano, E.: Simulation of Vibratory Conveyance of Granular Materials Using Discrete Element Method. *Transactions of the Japan Society of Mechanical Engineers, Series C* 64(625), 3264–3270 (1998)
5. Kawaguchi, T., Tanaka, T., Tsuji, Y.: Numerical Simulation of Fluidized Bed using the Discrete Element Method (the Case of Spouting Bed). *Transactions of the Japan Society of Mechanical Engineers, Series C* 58(551), 2119–2125 (1992)
6. Taguchi, Y.: New Origin of a Convective Motion: Elastically Induced Convection in Granular Materials. *Physical Review Letters* 69(9), 1367–1370 (1992)
7. Yoshida, J., Gotoh, K., Masuda, H.: A Study on Size Segregation of Particles Using Distinct Element Analysis. *Journal of Chemical Engineering of Japan* 22(3), 622–628 (1996)
8. Sakaguchi, H., Murakami, A., Hasegawa, T., Shirai, A.: Connected Lattice Cellular-Automaton Particles: A Model for Pattern Formation in Vibrating Granular Media. *Soils and Foundations* 36(1), 105–110 (1996)
9. Baxter, G.W., Behringer, R.P.: Cellular Automata models for the flow of granular materials. *Physica D* 51, 465–471 (1991)
10. Nakano, T., Miyamoto, S., Morishita, S., Sato, Y.: Particle Flow Simulation by Cellular Automata Method. *Transactions of the Japan Society of Mechanical Engineers, Series C* 64(617), 134–140 (1998)
11. Komatsuzaki, T., Sato, H., Iwata, Y., Morishita, S.: Modeling of Granular Damper using Cellular Automata. *Transactions of the Japan Society of Mechanical Engineers, Series C* 69(685), 2280–2286 (2003)

Convex Hulls on Cellular Automata

Luidnel Maignan¹ and Fredric Gruau^{1,2,3}

¹ Institut National de Recherche en Informatique et Automatique, Centre de Saclay,
Parc Orsay Université, 4 rue J. Monod, 91893 Orsay Cedex, France

² Laboratoire de Recherche en Informatique

Bât 490, Université Paris-Sud 11, 91405 Orsay Cedex, France

³ Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier
Université Montpellier 2, 161 rue Ada, 34095 Montpellier Cedex 5, France

Abstract. In the cellular automata domain, the discrete convex hull computation rules proposed until now only deal with a connected set of seeds in infinite space, or with distant set of seeds in finite space. We present a cellular automata rule that constructs the discrete convex hull of arbitrary set of seeds in infinite spaces. The rule is expressed using intrinsic and general properties of the cellular spaces, considering them as metric spaces. In particular, this rule is a direct application of metric Gabriel graphs. This allows the rule and its components to be used on all common 2D and 3D grids used in cellular automata.

1 Introduction

In abstract convexity theory [10,13], a *convexity* $\mathcal{C} \subset 2^S$ over a set S is a collection of subsets of S , called *convex subsets* or *convex sets*, that is closed under intersection. The *convex hull* $H_{\mathcal{C}}(S_0)$ of an arbitrary subset $S_0 \subset S$, whose elements will be called *seeds*, is the minimal convex set containing S_0 . For specific S and \mathcal{C} depending on the domain of application, finding an algorithm that realizes the convex hull operator $H_{\mathcal{C}}: 2^S \rightarrow \mathcal{C}$ is a common problem.

The most known and studied convexity is defined over Euclidean spaces \mathbb{R}^d . A subset is *Euclidean-convex* if it contains all the segments joining two of its points. For example, Fig. 1(a) shows a non Euclidean-convex set, commonly called concave set, along with two segments that are not contained in the set. In Fig. 1(b), it is not possible to find any missing segment. Figure 1(c) gives an example of convex hull. Algorithms that construct Euclidean-convex hulls for different dimensionality include Jarvis's gift wrapping [8], Graham's scan [6], Kirkpatrick–Seidel algorithm [9], and Chan's algorithm [3].

Euclidean convexity is a particular case of *metric convexity*. This latter is defined over metric spaces, i.e. sets of points associated with a distance function (also called *metric*). A subset is *metric-convex* if it contains all the shortest paths joining two of its points. In domains like computer graphics or cellular automata, the metric spaces typically consist of a set of pixels or cells, the length of a path being related to the number of pixels or cells lying on it. Networks of computers, wireless devices or sensors consider the metric space corresponding to the graph

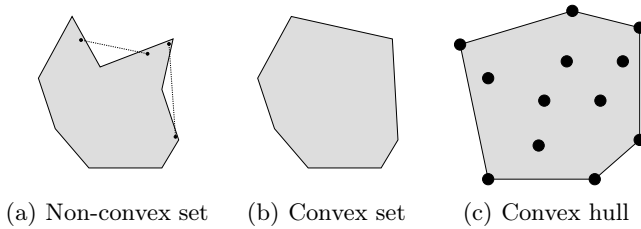


Fig. 1. Euclidean convexity examples. Considered set for convexity in gray, seeds in black. Note that (b) is also the convex hull of (a).

of communication, each edge having an arbitrary length, and the length of a path being the sum of the lengths of its edges. More abstract metric spaces are considered in machine learning and multivariate analysis for example.

In this article, we tackle the convex hull problem in the cellular automata framework [7]. In this parallel computation framework, the set of processing elements themselves are the points of the space and can only communicate locally. The problem is then to have each processing element to compute the convex hull of a selected subset of them by selecting itself if it is in it. We start by describing the cellular automata framework, and the metric spaces and convexities related to it in Sect. 2. We review the existing results in this domain and compare them with ours in Sect. 3. We describe our solution incrementally, by giving the intuition that leads to it in Sect. 4.

2 Framework and Definitions

2.1 Cellular Automata Framework

A cellular automaton is made of an infinite set S of *sites*, i.e. processing elements having a particular position in space. S is called the *cellular space* and forms a crystalline graph, whose neighborhood relation is denoted as N . Being processing elements, each site x has a particular state $v_t(x) \in V$ (as value) that changes with time, V being finite. The updating of the sites is synchronous and local. This means that all the sites update their states at the same time using the same updating rule. The latter determines the next state $v_{t+1}(x)$ of each site x based on the previous state of a finite number of its closest sites. In this framework, the convex hull problem is formulated this way: Given a set of seeds S_0 , represented by a configuration src defined as $src(x) \equiv x \in S_0$, the goal is to find a rule R such that, from some instant t , $R_t(x) \equiv x \in H_C(S_0)$. Examples of initial and associated final configuration for different grids are given in Fig. 4.

2.2 Cellular Spaces and Metric Spaces

Different cellular spaces may be used and, regarding convexities, different metrics can be associated to them. The most commonly used bi-dimensional cellular

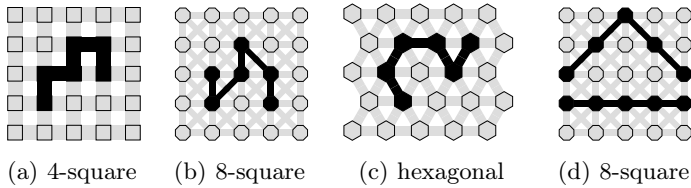


Fig. 2. Grids used in this article: the polygons (squares, octagons and hexagons) correspond to the sites, the lines correspond to the edges, and example paths are in black.

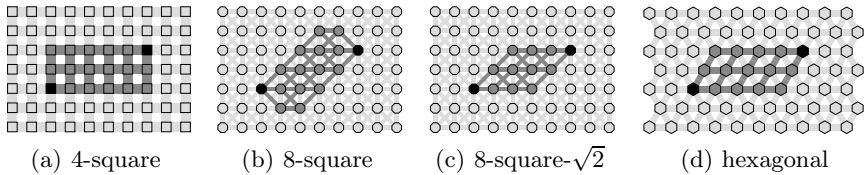


Fig. 3. Intervals for different grids

spaces are the square grids, with 4 or 8 neighbors per site, and the hexagonal grids, having 6 neighbors per site. Figure 2 shows these grids, exposing the nodes and the edges.

For the distance function, we consider the *hop count metric*, which associates to each edge the unitary length. This is a natural choice for the 4-square and hexagonal grids since all edges have the same length when we draw them. However, this is not the case for the 8-square grids, whose diagonal edges are drawn $\sqrt{2}$ times longer than the vertical and horizontal ones. Therefore, we also consider two metrics for the 8-square grids: the hop count metric, and the $\{1, \sqrt{2}\}$ metric that associates unitary and $\sqrt{2}$ lengths to non-diagonal and diagonal edges respectively. With the hop count metric, paths represented on Figs. 2(a), 2(b), and 2(c) have length 5. With the $\{1, \sqrt{2}\}$ metric, the path of Fig. 2(b) has length $3 + 2\sqrt{2}$. It is important to note that paths of Fig. 2(d) have the same length for hop count and different lengths for $\{1, \sqrt{2}\}$, since this has an effect on the notions of shortest path and convexity.

2.3 Convexities and Convex Hulls

For an arbitrary metric space, a point z lying on a shortest path joining two points x and y is said to be *between* the two points. It is denoted as $z \in [x, y]$, where $[x, y]$ is called the *interval* between x and y . Formally, we have $[x, y] = \{z \in S \mid d(x, z) + d(z, y) = d(x, y)\}$ where d is the metric and S the set of points. In Euclidean spaces, $[x, y]$ corresponds to the segment joining the points x and y , since it is the unique shortest path joining these points. In other metric spaces, including square and hexagonal grids, there may be many shortest paths between two points, leading to more complex intervals as shown in Fig. 3.

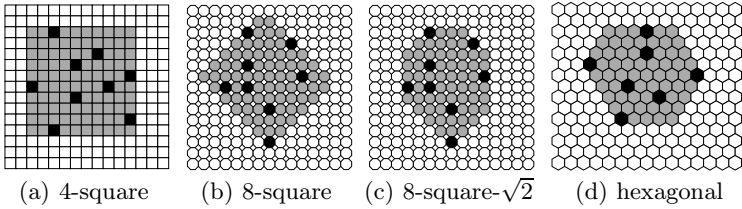


Fig. 4. Each site x is black if $x \in S_0$, or gray if $x \in H_C(S_0)$

Using intervals, metric convexity can be defined as follows. A subset $C \subseteq M$ is metric-convex if and only if $[x, y] \subseteq C$ for any pair of points $(x, y) \in C^2$. If we define the operator $I(P) = \bigcup \{ [x, y] \mid (x, y) \in P^2 \}$ that adds to a set of points the shortest path joining them, we can say that a set C is metric-convex if and only if $I(C) = C$.

The convex hull $H_C(S_0)$ of a set $S_0 \subset S$ of seeds is the minimal convex set containing S_0 . Using $I(\bullet)$, it corresponds to the limit of the sequence $I(S_0), I(I(S_0)), \dots$, since $I(\bullet)$ adds points that have to be in the convex set, and the limit L verifies $I(L) = L$. Our algorithm also adds points iteratively.

3 State of the Art

The research about convex hulls on cellular automata mainly studies 8-square grids with $\{1, \sqrt{2}\}$ metric. In fact, the convexity is often not defined using sets, as we did, but using angles, and intersecting half planes. Although the definitions are distinct, the convex hulls are the same objects. For example, the 45-convexity is a version of the Euclidean convexity that only allows lines having angles of multiples of 45 to be used on the boundary of the convex hull. In this framework, the four metric spaces described previously correspond to the set of angles $\{0, 90\}$, $\{45, 135\}$, $\{0, 45, 90, 135\}$ and $\{0, 60, 120\}$.

3.1 Rule for Connected Set of Seeds

In [1], some of the first proposed rules for the convex hull problem are described. The intuition is that any bordering site that does not have a local configuration corresponding to the shape of a convex set boundary has to select itself. After observing that all rejected local configurations have 1, 2 or 3 selected sites, the rule is simplified to a counter that checks whether there are at least 4 selected sites in the neighborhood. This rule can be generalized to the majority rule, since 4 can be interpreted as the half of 8, the number of neighbors. The convex hull behavior of the majority rule is described in [7]. In fact, the set of seeds does not need to be connected, but simply denser enough, since only their quantity matters. Also, as more and more sites are selected by the rule, many local convex hulls can merge to form bigger convex hulls that can also merge, etc. Using our

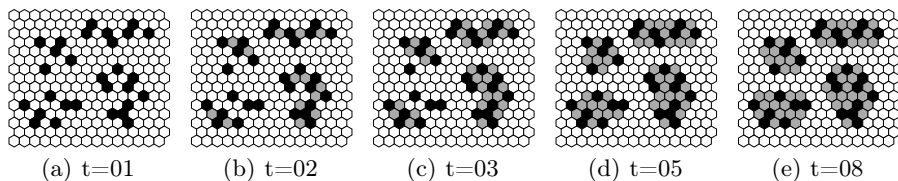


Fig. 5. With the set of seeds (a), we obtain a set of convex hulls

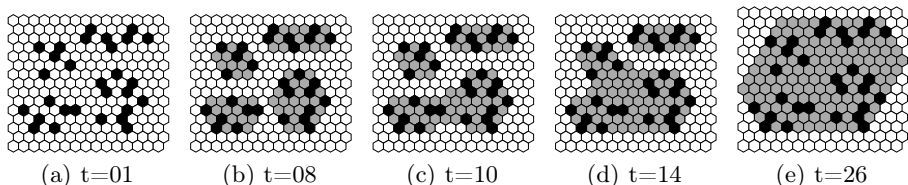


Fig. 6. Shifting only one seed to its neighbors connects recursively all the hulls

notations, the majority rule can be expressed as follows, and gives the results shown in Figs. 5 and 6 for hexagonal grids.

$$\text{majority}_{t+1}(x) = \begin{cases} \top & \text{if } \text{src}(x) \\ \top & \text{if } \text{card}\{y \in N(x) \mid \text{majority}_t(y)\} \geq \frac{\text{card}(N(x))}{2} \\ \perp & \text{otherwise.} \end{cases}$$

3.2 Rule for Already Wrapped Seeds

In [14, 4], a rule is proposed for non connected set of seeds. However, it requires the seeds to be finitely wrapped in a connected and compact pattern. One can also consider that this rule only works for finite spaces, considering the space itself as the finite wrapper. The proposed rule consists of two globally successive stages. The first one erodes the wrapper until a minimal isometric set is obtained. Thus between any pair of points of the set, at least one shortest path is in the set. The second stage is the application of a rule to transform this connected set of sites into its convex hull, as done in the previous subsection. The stages are shown in Fig. 7.

3.3 Comparison with Our Rule

In comparison with this approach, we do not require any bound on the space, which can therefore be infinite. As a result, convergence of our cellular automaton takes an infinite time, as there can be sites infinitely far from the seeds. Still, each site converges locally, and the convex hull itself will be built in finite time, if the distances between nearest seeds are upper bounded.

Our rule does not work directly for $\{1, \sqrt{2}\}$ metrics, because we manipulate distances that have to be integers in order to be encoded using finite states.

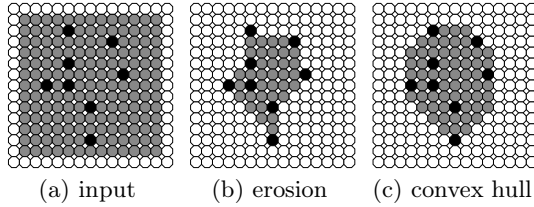


Fig. 7. Stages from wrapped seeds to their convex hull: The initial wrapping (a) is shrunk to (b) and is then grown to convex hull (c)

However, this is not a problem, since we can still produce the $\{1, \sqrt{2}\}$ convex hull having both diagonal and vertical-horizontal border, by intersecting two convex hulls: the 4-square one, having only vertical-horizontal borders, and the 8-square one with hop count, having only diagonals.

The formulation of our rule does not use boundaries at all, which has an important consequence for generalization: the rule applies directly to many bidimensional grids, including the ones listed in Fig. 2. It also works for their tridimensional counterparts. Lastly, it can be easily applied to bigger neighborhoods, resulting in faster convergence when needed.

4 Rules for Arbitrary Set of Seeds Using Hop Count

4.1 Rule for Local Convexity

Computing the convex hull locally means that each site has to select itself if it belongs to the convex hull of the selected sites present in its neighborhood. The computation of the local convex hull is an easy task due to the simplicity and finiteness of the space in the neighborhood of each site. Indeed, it is enough for a site to check if it lies on a shortest path joining two of its selected neighbors. This gives the following local convexity rule:

$$conv_t(x) = \begin{cases} \top & \text{if } src(x) \\ \top & \text{if } \exists \{y_0, y_1\} \subset \{y \in N(x) \mid conv_{t-1}(y)\}; x \in [y_0, y_1] \\ \perp & \text{otherwise.} \end{cases}$$

Let us mention that testing $x \in [y_0, y_1]$ with hop count metric is equivalent to testing $d(y_0, y_1) = 2$ since $x \in [y_0, y_1] \Leftrightarrow d(y_0, x) + d(x, y_1) = d(y_0, y_1)$ and $y \in N(x) \Leftrightarrow d(x, y) = 1$. For the $\{1, \sqrt{2}\}$ metric, the test has to be done explicitly.

Because it follows directly from the convex hull definition, this rule is more precise than the *majo* rule. It is possible to check that whenever one selects half of the neighbors of a site x , then two of them are joined by a shortest path containing the site x . The reverse is obviously not true, which means the

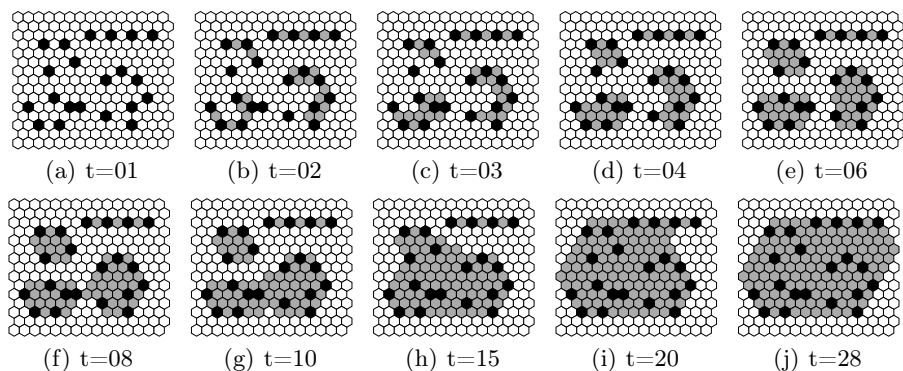


Fig. 8. Evolution of the *conv* rule. The *majo* rule is stationary on (a).

conv rule is able to construct the convex hull in more cases than the *majo* rule. One can also note that applying *majo* on an 8-square grid can only give a $\{1, \sqrt{2}\}$ -convex hull. The formulation of *conv* allows choosing the metric to use, and tackles many grid topologies at once, such as the ones considered in this article and their tridimensional counterparts. However, it exhibits roughly the same behavior. Figure 8 shows the evolution of the *conv* rule with an initial configuration on which *majo* is stationary.

4.2 Global Convexity with Only Two Seeds

Since we have seen solutions to transform a connected set of seeds into its convex hull, a natural idea to obtain the convex hull of an arbitrary set of seeds is to connect them in a minimal way that remains in the desired convex hull. We start by studying the simpler case of only two seeds. In this setting, the goal is to select sites of the interval, as shown in Fig. 3.

To do so, we compute the distance of each site to the nearest seed, and look at the resulting pattern (Fig. 9(a)). We can notice distinct sites, namely the middle sites which are exactly the middle of the shortest path joining the two seeds. It turns out that these middles can always be detected by looking at the distance values in a bounded neighborhood. Therefore, they will be the first sites identified as being in the convex hull (Fig. 9(b)). All the other sites of the interval can then be selected by back-propagating from the middles to the seeds, by traveling towards neighbors that are closer to the seeds, again by using the distance values (Fig. 9(c)). This achieves the desired construction, without using any global phase transition but only local interaction. Let us now describe each rule in more details.

Distance Field. The distance computation described earlier is what we call a *distance field*. For hop count metrics, it associates to each site an integer and

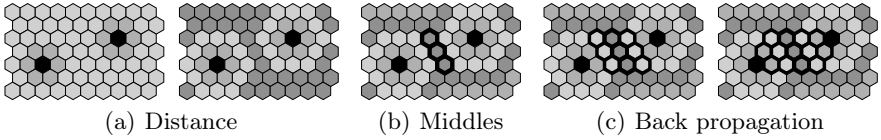


Fig. 9. The rules *dist*, *cent* and *back* in action

can be expressed by the following rule. The latter converges to $dist_\infty(x) = \min\{d(x, y) \mid src(y)\}$:

$$dist_t(x) = \begin{cases} 0 & \text{if } src(x) \\ 1 & \text{if } \neg src(x) \wedge t = 0 \\ \min\{1 + dist_{t-1}(y) \mid y \in N(x)\} & \text{otherwise.} \end{cases}$$

While this rule has an infinite number of possible state, we only need its gradient, i.e. the differences between the distance of neighboring sites. In [11], we have shown how to represent the gradients of some kind of integer fields with a finite number of states, thanks to the modulo operator. Applied on the *dist* rule, it allows representing it modulo 3 and gives the way to compute the gradient from these modulo values. For brevity and readability, we use directly *dist* in the rest of the paper and redirect the interested reader to [11]. Finally, we do not have any mean to represent the distance field for $\{1, \sqrt{2}\}$ metric with finite number of states, which is the reason why the final rule can not be directly applied to this metric.

From Middles Back to Seeds. As mentioned earlier, the middle sites are detected using the distance values present in their neighborhood. The global idea is to detect distance patterns that only happen when there are two nearest seeds for the sites, such that the site is between them. Because we treat this detection in our general framework instead of a particular grid, we delay the discussion about this detection to the next subsection, and directly use $cent(x)$ to denote that a site x is a middle.

For the back-propagation, each site having a selected site in its neighborhood has to determine if it is closer or not than the selected neighbor. If it is so, it can select itself, since it is between the selected neighbor and the seeds. This is expressed as:

$$back_t(x) = \begin{cases} \top & \text{if } cent_t(x) \\ \top & \text{if } t > 0 \wedge \exists y \in N(x), back_{t-1}(y) \wedge dist_{t-1}(x) < dist_{t-1}(y) \\ \perp & \text{otherwise} \end{cases}$$

4.3 Global Convexity and Metric Gabriel Graphs

When considering the general case with many seeds, some questions naturally arise: do the rules presented for only two seeds do all the work pairwise? Do they

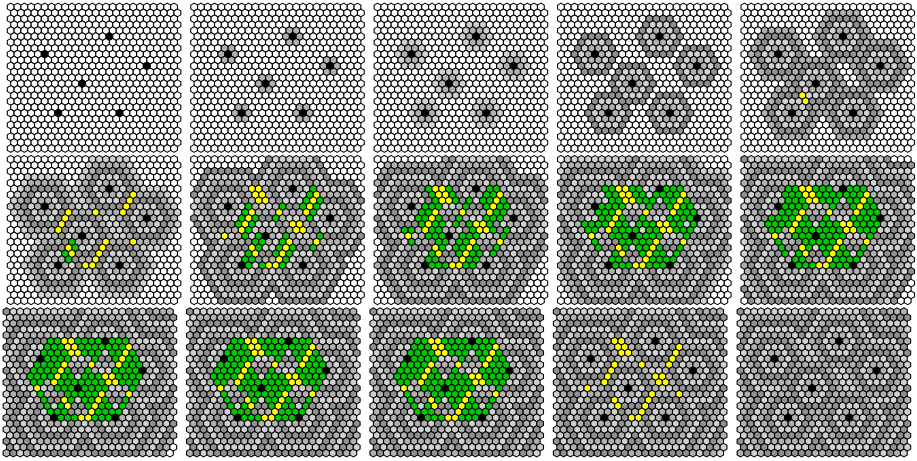


Fig. 10. Snapshots of the computation of the three layers simultaneously. The two last snapshot show the final configuration with, firstly *back* hidden and then *back* and *cent* hidden. Red: generators, gray: *dist*, yellow: *cent*, green: *back*.

produce a connected set? What structure is constructed? The answer is that we produce a connected set, connecting the seeds pairwise to draw a structure related to Delaunay graphs. A complete description is beyond the scope of this article but can be found in [12]. We only give here the material that allows understanding the global structure of the constructed graph.

When computing a distance field, one implicitly associates to each site its nearest seed. It is strongly related to Voronoi diagram [2], the set of sites having the same nearest seed being called the Voronoi region of the seed and the sites having many nearest seeds being the boundaries between the Voronoi regions. In our case, we detect boundary sites that are on a shortest path between the corresponding seeds, to make sure to select only sites that are in the convex hull.

By doing so, we only connect seeds of neighboring Voronoi regions, such that there is a shortest path going through the boundary between the two regions. Replacing the words “shortest path” by “segment”, we obtain one of the properties of Gabriel graphs [5], a connected sub-graph of the Delaunay graph defined for Euclidean spaces. In [12], we generalize the definition of Gabriel graphs to arbitrary metric spaces and obtain *metric Gabriel graphs*, which identify exactly what we need to detect in order to have a connected set of sites. We also explain in detail the rule *cent* (as metric Gabriel ball *centers*).

By using metric Gabriel graphs, we have, roughly speaking, that $back \circ cent \circ dist$ constructs a connected set of sites that is a subset of the convex hull. In order to complete the convex hull, we simply have to consider $conv \circ back \circ cent \circ dist$. The final cellular automaton thus described has 7 states: (3 distance states) * (2 “in convex hull” states) + 1 special “seed” state. Because of *cent* rule, it uses a neighborhood of radius 2. The evolution of the rule without *conv* is shown in Fig. 10.

Acknowledgments

We would like to thank Prof. Adamatzky who pointed us first to the convex hull problem, which was the starting point of this work, and secondly to Gabriel graphs, when we showed him our particular graph and its properties.

References

1. Adamatzky, A.: Computing in nonlinear media and automata collectives. IOP Publishing Ltd., Bristol (2001)
2. Aurenhammer, F.: Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Comput. Surv.* 23(3), 345–405 (1991)
3. Chan, T.M.: Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry* 16, 361–368 (1996)
4. Clarridge, A.G., Salomaa, K.: An improved cellular automata based algorithm for the 45-convex hull problem. *Journal of Cellular Automata* (2008)
5. Gabriel, R.K., Sokal, R.R.: A new statistical approach to geographic variation analysis. *Systematic Zoology* 18(3), 259–278 (1969)
6. Graham, R.L.: An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters* 1(4), 132 – 133 (1972), [http://dx.doi.org/10.1016/0020-0190\(72\)90045-2](http://dx.doi.org/10.1016/0020-0190(72)90045-2)
7. Ilachinski, A.: Cellular Automata: A Discrete Universe. World Scientific Publishing Co., Inc., River Edge (2001)
8. Jarvis, R.A.: On the identification of the convex hull of a finite set of points in the plane. *Information Processing Letters* 2(1), 18 (1973), [http://dx.doi.org/10.1016/0020-0190\(73\)90020-3](http://dx.doi.org/10.1016/0020-0190(73)90020-3)
9. Kirkpatrick, D.G., Seidel, R.: The ultimate planar convex hull algorithm? *SIAM Journal on Computing* 15(1), 287–299 (1986), <http://link.aip.org/link/?SMJ/15/287/1>
10. Levi, F.: On helly’s theorem and the axioms of convexity. *J. of Indian Math. Soc.*, 65–76 (1951)
11. Maignan, L., Gruau, F.: Integer gradient for cellular automata: Principle and examples. In: *IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, pp. 321–325 (2008)
12. Maignan, L., Gruau, F.: Gabriel graphs in arbitrary metric space and their cellular automaton for many grids. *ACM Trans. Auton. Adapt. Syst.* (2010) (in press)
13. Singer, I.: *Abstract convex analysis*. John Wiley & Sons Inc., Chichester (1997)
14. Torbey, S., Akl, S.G.: An exact and optimal local solution to the two-dimensional convex hull of arbitrary points problem. *Journal of Cellular Automata* (2008)

Square Kufic Pattern Formation by Asynchronous Cellular Automata

Seyyed Amir Hadi Minoofam¹ and Azam Bastanfard²

¹ Computer Engineering Faculty, Islamic Azad University-Qazvin Branch,
Daneshgah St., Nokhbegan Blvd., Qazvin, Iran

² Computer Engineering Faculty, Islamic Azad University-Karaj Branch,
Imam Ali Complex, Moazen Blvd., Karaj, Iran

minoofam@qiau.ac.ir, bastanfard@kiaiu.ac.ir

Abstract. Algorithms Design for pattern formation is an interesting science and many investigations are done about it. In this paper, a novel pattern formation method using asynchronous cellular automata for generating square Kufic patterns is explored. Square Kufic is a kind of script that is used in many Islamic architecture buildings. In our new method, cellular automata rules causing a specified kind of pattern are manually extracted. The implementation results show this fact by using some special patterns in replicators.

Keywords: Asynchronous Cellular Automata, Computer Graphics, Pattern Formation, Square Kufic script.

1 Introduction

Cellular automata are dynamical systems such that the current state of each cell is determined by its neighborhood state at the previous time step. We consider in this paper an interactive particle system modeled by a cellular automaton having two kinds of cells, occupied and empty cells.

The main contribution of this paper is using asynchronous cellular automata to generate script patterns. These particular patterns are used in cultural heritage. Square Kufic patterns are based on grids and the best way for implementing such patterns is a grid based method like cellular automata [1]. Using cellular automata has several benefits. For instance, it is simple because it is based on square grids like the forming elements of square Kufic script. Also it is scalable and flexible.

The rest of the paper is organized as follows. Section 2, describes the basic notions of asynchronous two dimensional cellular automata. Section 3, provides the rule set for generating some patterns. Experimental results appear in section 4. The paper is concluded in section 5.

2 Asynchronous Cellular Automata

In general, it has been found that the asynchronous cellular automata evolve much differently from their synchronous counterparts [2]. There are various methods to

update cells asynchronously and these fall into two distinct categories, step-driven and time-driven. In this paper, we use the latter method.

This paper uses extended Moore neighborhood in its simplest form. It simply handles joint and disjoint letters. In some literatures the beginning letters of the direction names are used to refer neighbor cells. For example "C" points to the internal cell and "NE" refers to the northeast cell. We extend this method of naming to refer to the cells in an easy understanding way [3]. For instance, "NEE" refers to the northeast east cell, which means one hop to north and then two hops toward east direction.

Next section explains about transition rules for generating square Kufic patterns.

3 Rule Extraction

Each cell's state is assumed to take an integer equal either to nil or unity. The nil state denotes an empty cell having white color, and the other state is one occupied by a particle having black color.

Figure1 (a) and (b) show two architectural buildings contain two models of "Allah" pattern.

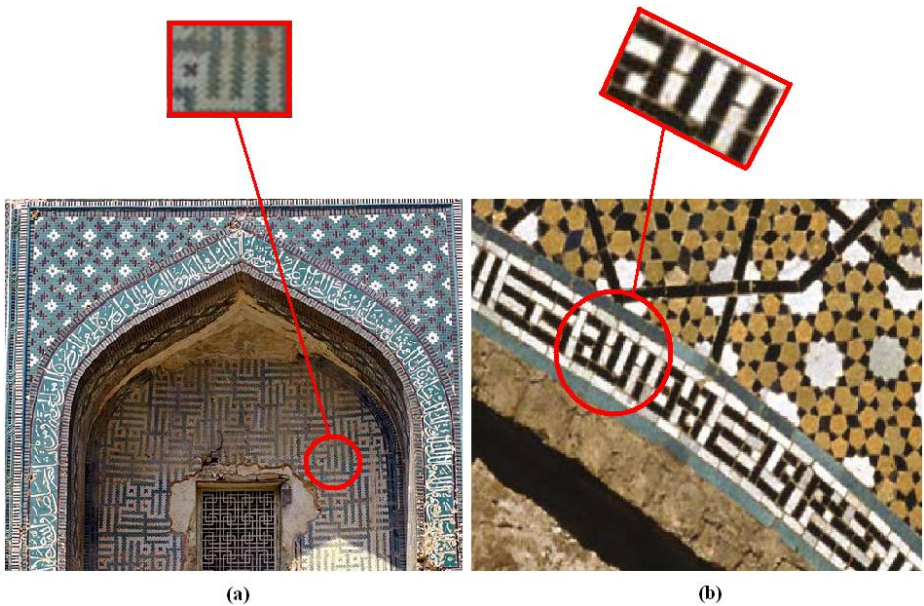


Fig. 1. (a) Model 1 of "Allah" pattern tiling from the Tughlugh Timur Mausoleum of Xinjiang. (b) Model 2 of "Allah" pattern tiling from the Imam Mosque of Isfahan.

First of all, the patterns have to decompose in to their constructing letters. Then, the transition rules set for each letter must be extracted manually. After that, the cellular automata can produce the whole of each pattern asynchronously.

In each rule if all the mentioned cells are black, then the rule is true, and it causes an effect in cellular automata at the next time step. Otherwise, if even one of the mentioned cells isn't black, then the rule is false which causes no effect on the cellular grid.

Figure 2 demonstrates the asynchronous process for generating patterns respectively.

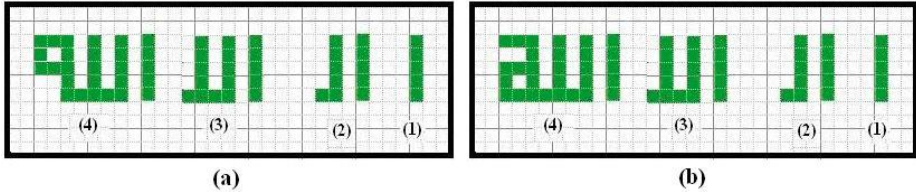


Fig. 2. (a) Generating process for model 1 of "Allah" pattern. (b) Generating process for model 2 of "Allah" pattern.

Consequently, it is possible to provide a flexible algorithm for generating these patterns. Reused rules can make the algorithm shorter and simpler too [4].

In this step, we define a transition function φ that operates on the rules with xor " \oplus " operation.

$$\varphi[R_1, R_2, \dots, R_n] = \bigoplus_{i=1}^n R_i \tag{1}$$

Relation (1) describes that all rules R_i should be xor (\oplus) with each other to introduce the correct results. Note that for each letter, some functions φ with different R_i s are existing. This Algorithm is the determination of each transition that is a mapping from a neighborhood pattern to another neighborhood pattern. We at first set all of the transitions to be "inactive": Any patterns of neighborhood are not changed by the update. In the following we will replace some inactive transitions with "active" ones that change the pattern of neighborhood by the update [5]. Note that, for the simplicity sake, in this section we were considering only a few set rules.

The next section demonstrates some results that are obtained using above algorithm.

4 Results

The results of our algorithms which are implemented in Visual C++ environment using OpenGL library are illustrated as follows. An example of executing on model 1 of "Allah" pattern is illustrated in figure 3 (a). In figure 6 (b), the execution of relation on model 2 of this pattern is shown. Result of executing relation on model 1 of "Allah" pattern is illustrated in figure 3 (c). In figure 3 (d), the execution on model 2 of this pattern is demonstrated.



Fig. 3. (a) Applying on model 1 of "Allah" pattern after 40 steps. (b) Applying on model 2 of "Allah" pattern after 24 steps. (c) Applying on model 2 of "Allah" pattern after 24 steps. (d) Applying on model 1 of "Allah" pattern after 24 steps.

5 Conclusion

Using asynchronous cellular automata leads to efficient implementation of the algorithm. Because transition rules become active in a limited time steps, the algorithm is very simple. Also, various patterns can be generated without any rule confliction. Moreover, common parts in each letter will be formed by using the same rules. A comparison between this paper algorithm and other algorithms may be expected, but no other method was investigated before. Therefore, this is the first algorithm for generating square Kufic patterns by asynchronous CA.

The Square Kufic script is a good instance in calligraphy which provides different versions of writing holy words. As this script is based on a square grid, the proposed algorithm not only provides the simplicity and flexibility for generating these patterns but also has the power for controlling complexity and possibility of extending the functionality of the rules. Examples of Square Kufic script can be found in cultural heritage and old literatures. Also it is so popular in modern architectures, handicrafts, decorations and art. One of our future works is designing more complex patterns and searching about making use of geometrical transformations in cellular automata.

References

1. Steeb, W.H.: The Nonlinear Workbook. World Scientific Publishing, Singapore (2005)
2. Beigy, H., Meybodi, M.R.: Asynchronous Cellular Learning Automata. *J. Auomatica* 44(4), 1350–1357 (2008)
3. Minoofam, S.A.H., Bastanfard, A.: A Novel Algorithm for Generating Muhammad Pattern Based on Cellular Automata. In: 13th WSEAS International Conference on Applied Mathematics (MATH 2008), pp. 339–344 (2008)
4. Mitra, S., Kumar, S.: Fractal Replication in Time-manipulated One Dimensional Cellular Automata. *J. Complex Systems* 16(3), 191–207 (2006)
5. Schiff, J.L.: Cellular Automata: A Discrete View of the World. John Wiley, Chichester (2008)

Development and Calibration of a Preliminary Cellular Automata Model for Snow Avalanches

Maria Vittoria Avolio¹, Alessia Errera², Valeria Lupiano³,
Paolo Mazzanti^{4,5}, and Salvatore Di Gregorio¹

¹ Univ. of Calabria, Dept. of Mathematics & Center of High-Performance Computing,
Arcavacata, 87036 Rende (CS), Italy
{avoliomv,dig}@unical.it

² eni, exploration & production division, via Emilia 1, 20097 San Donato Milanese, Italy
alessia.errera@eni.com

³ Univ. of Calabria, Dept. of Earth Sciences, Arcavacata, 87036 Rende (CS), Italy
lupianov@unical.it

⁴ NHAZCA S.r.l., spin-off “Sapienza Università di Roma”, Via Cori snc, 00177, Roma, Italy
paolo.mazzanti@nhazca.com

⁵ Univ. of Rome “Sapienza”, Dept. of Earth Sciences, P.le Aldo Moro, 00185, Roma, Italy
paolo.mazzanti@uniroma1.it

Abstract. Numerical modelling is a major challenge in the prevention of risks related to the occurrence of catastrophic phenomena. A Cellular Automata methodology was developed for modelling large scale (extended for kilometres) dangerous surface flows of different nature such as lava flows, pyroclastic flows, debris flows, rock avalanches, etc. This paper presents VALANCA, a first version of a Cellular Automata model, developed for the simulations of dense snow avalanches. VALANCA is largely based on SCIDDICA-SS2, the most advanced model of the SCIDDICA family developed for flow-like landslides. VALANCA adopts several of its innovations: outflows characterized by their mass centre position and explicit velocity. First simulations of real past snow avalanches occurred in Switzerland in 2006 show a satisfying agreement, concerning avalanche path, snow cover erosion depth and deposit thickness and areal distribution.

Keywords: Cellular Automata, Modelling and Simulation, Snow Avalanche.

1 Introduction

Snow avalanches are rapid gravity-driven movements of snow masses down mountain slopes. They may be included in the category of granular flows together with mudflows, debris flows, pyroclastic flows and rock avalanches. In fact, there is experimental evidence for snow avalanches exhibiting all the flow regimes identified in granular flows, from the quasi-static to the collisional, grain-inertia and macroviscous regimes [1].

Dense avalanches have a high density core ($100\text{--}500\text{ kg/m}^3$) at the bottom with particle sizes from 1 mm to 1 m, typical flow depths between 0.5 and 5 m and velocities in the range 5–40 m/s. They are a manifestation of the quasi-static and

collisional regimes. On the other extreme, powder snow avalanches are dilute flows of small snow particles (< 1 mm) suspended in the air by intense turbulence. The density is much lower than in dense avalanches (typically $1\text{--}10\text{ kg/m}^3$), but the flow depth ($10\text{--}100$ m) and average velocity ($30\text{--}100$ m/s) are much larger. In recent years, the important role of the fluidised regime, intermediate between these two end members, has been recognised ([1], [2], [3], [4]). Typical densities and flow velocities are $10\text{--}100\text{ kg/m}^3$ and $30\text{--}70$ m/s, respectively.

The urgent and increasing need for protection of settlements and traffic routes from snow avalanches has led to several approaches for modelling avalanches over the past 90 years [5].

There is a wide variety of fluid mechanics-based models; they differ in complexity and also with regard to the type of avalanche they describe [5]. Dense snow avalanches can be described by mass-point models, e.g. [6], or continuum models based on the Navier–Stokes or Saint-Venant equations, with a constitutive equation appropriate for flowing snow. In the case of powder snow avalanches and slush flows, it may be necessary to use binary mixture theory to describe the dynamics of the particles and the interstitial fluid satisfactorily [7]. Models of the Saint-Venant type exploit that snow avalanches (and in particular dense avalanches) are shallow flows by integrating the balance equations of mass and momentum (and energy) over the direction perpendicular to the ground ([5], [8], [9]) for more details and references to the original works.

A different approach, based on the computational paradigm of Cellular Automata, was adopted by Barpi et al. [10], that developed the model ASCA (cf. Section 2.2) for the simulation of snow avalanches. ASCA simulations of avalanches, that occurred in Susa Valley (Western Italian Alps), were able to reproduce the correct three-dimensional avalanche path and the order of magnitude of the avalanche deposit.

Kronholm et al. [11], instead, used a Cellular Automata based model to show how the spatial structure of shear strength may be critically important for avalanche fracture propagation.

A Cellular Automata (CA), at the basis of the model presented in this work, evolves in a discrete space-time. Space is partitioned in cells of uniform size, each cells embeds a Finite Automaton (FA) computing unit, that changes the cell state according to the states of the neighbour cells, where the neighbourhood conditions are determined by a pattern invariant in time and space [12]. An extension of classical CA [12] was developed in order to model many complex macroscopic fluid-dynamical phenomena, that seem difficult to be modelled in other CA frames (e.g. the lattice Boltzmann method), because they take place on a large space scale.

Such CA can need a large amount of states, that describe properties of the cells (e.g. temperature); such states may be formally represented by means of sub-states, that specify the characteristics to be attributed to the state of the cell and determining the CA evolution. It involves a large amount of states more a complicated transition function, not reducible to a lookup table.

In the case of surface flows, quantities concerning the third dimension, i.e. the height, may be easily included among the CA sub-states (e.g. the altitude), permitting models in two dimensions, working effectively in three dimensions. Furthermore, an algorithm for the minimisation of the differences (in height) [12], [13] was found in this context in order to determine the outflows from a cell toward the remaining cells

of its neighbourhood, giving rise to several models for different macroscopic phenomena: lava flows [12], debris/mud flows [12] and rain soil erosion [14].

Explicit velocity solution is adopted: moving flows toward the neighbouring cells are individuated by the sub-states mass, velocity and mass centre co-ordinates. The resulting new mass, mass centre and velocity are computed by composition of all the inflows from the neighbours and the residual quantities inside the cell [15], [16].

This paper illustrates VALANCA (it is the Sicilian word for avalanche and acronym for “Versatile model of Avalanche propagation by LAws and Norms of Cellular Automata”), a new model for the simulation of snow avalanches. VALANCA profits of studies of Barpi et al. [10] but it included new features [13] of SCIDDICA-SS2, ([15], [16]), the most advanced model of the SCIDDICA family for flow-like landslides, developed by some authors of this paper. Some differences, with respect to the ACSA model by Barpi et al., are presented in Section 2.2.

The next section defines the model VALANCA, while the simulation results of two snow avalanches in Davos (Switzerland) are shown in the third section.

2 The Model VALANCA

VALANCA is a two-dimensional CA with hexagonal cells, the state of cell is specified by sub-states, the transition function is constituted by local “elementary” processes, applied sequentially:

$$\text{VALANCA} = \langle R, X, S, P, \tau \rangle$$

where

- R is the set of regular hexagons covering the region, where the phenomenon evolves.
- X identifies the geometrical pattern of cells, which influence any state change of the central cell: the central cell (index 0) itself and the six adjacent cells (indexes 1,...,6).
- S is the finite set of states of the finite automaton, embedded in the cell; it is equal to the Cartesian product of the sets of the considered sub-states:

$$S_A \times S_D \times S_{TH} \times S_X \times S_Y \times S_{KH} \times S_E^6 \times S_{XE}^6 \times S_{YE}^6 \times S_{KHE}^6 \times S_I^6 \times S_{XI}^6 \times S_{YI}^6 \times S_{KHI}^6$$
 - S_A is the cell altitude.
 - S_D is the snow cover depth, that could change into avalanche mass by erosion (Fig.1).
 - S_{TH} is the average thickness of avalanche mass inside the cell (Fig.1), S_X and S_Y are the co-ordinates of the mass centre with reference to the cell centre.
 - S_{KH} is the kinetic head of avalanche mass inside the cell (Fig.1)..
 - S_E is the part of avalanche mass, the so called “external flow”, (normalised to a thickness) that penetrates the adjacent cell from central cell, S_{XE} and S_{YE} are the co-ordinates of the external flow mass centre with reference to the adjacent cell centre, S_{KHE} is the kinetic head of avalanche mass flow. There are six components (one for each adjacent cell) for the sub-states $S_E, S_{XE}, S_{YE}, S_{KHE}$.
 - S_I is the part of avalanche mass toward the adjacent cell, the so called “internal flow”, (normalised to a thickness) that remains inside the central cell, S_{XI} and S_{YI} are the co-ordinates of the internal flow mass centre with reference to the central cell centre, S_{KHI} is the kinetic head of avalanche mass flow. There are six components (one for each adjacent cell) for the sub-states $S_I, S_{XI}, S_{YI}, S_{KHI}$.

- P is the set of the global physical and empirical parameters, which account for the general frame of the model and the physical characteristics of the phenomenon; the next section provides a better explication of the elements of the following set:

$$\{p_a, p_t, p_{fc}, p_{td}, p_{ed}, p_{mt}, p_{pe}\}$$

- p_a is the cell apothem;
- p_t is the temporal correspondence of a CA step;
- p_{fc} is the friction coefficient for avalanche outflows;
- p_{td}, p_{ed} are parameters for energy dissipation by turbulence and erosion;
- p_{mt} is the activation thresholds of the snow mobilisation;
- p_{pe} is the progressive erosion parameters;
- $\tau: S^7 \rightarrow S$ is the deterministic state transition for the cells in R . The basic elements of the transition function will be sketched in the next section.

At the beginning of the simulation, we specify the states of the cells in R , defining the initial CA configuration. The initial values of the sub-states are accordingly initialised. In particular, S_A assumes the morphology values; S_D assumes initial values corresponding to the maximum depth of the snow mantle cover except for the detachment area, where the thickness of the detached avalanche mass is subtracted from snowpack depth; S_{TH} is zero everywhere except for the detachment area, where the thickness of detached avalanche mass inside the cell is specified; all values related to the remaining sub-states are zero everywhere.

At each next step, the function τ is applied to all the cells in R , so that the configuration changes in time and the evolution of the CA is obtained.

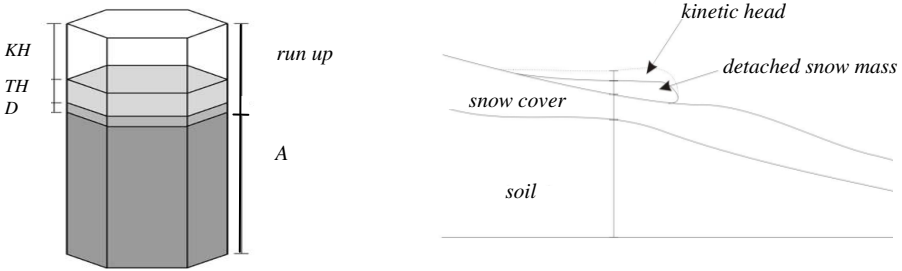


Fig. 1. Left: three-dimensions visualization of the sub-states S_A , S_D , S_{TH} and S_{KH} for a hexagonal cell. Right: an ideal vertical section of a snow flow.

2.1 The VALANCA Transition Function

Four local processes may be considered for VALANCA:

- snow cover, kinetic head and avalanche thickness variation by snow cover mobilisation;
- kinetic head variation by turbulence dissipation; avalanche outflows (height, mass centre co-ordinates, kinetic head) determination and their shift deduced by the motion equations;
- composition of avalanche mass inside the cell (remaining avalanche more inflows) and determination of new thickness, mass centre co-ordinates, kinetic head.

In the following, a sketch of the local elementary processes will be given, which is sufficient to capture the mechanisms of the transition function; the execution of an elementary process updates the sub-states. Variables concerning sub-states and parameters are indicated by their abbreviations in the subscripts. When sub-states need the specification of the neighbourhood cell, index is indicated between square brackets. ΔQ means variation of the value of the sub-state S_Q .

Mobilisation Effects. When the kinetic head value overcomes an opportune threshold $KH > mt$ depending on the snow cover features then a mobilisation of the snow cover occurs proportionally to the quantity overcoming the threshold: $pe \cdot (KH - mt) = \Delta TH = -\Delta D$ (the snow cover depth diminishes as the avalanche thickness increases), the kinetic head loss is: $-\Delta KH = ed \cdot (KH - mt)$. The mixing of the eroded snow cover with the earlier avalanche mass involves that the earlier kinetic energy of avalanche mass becomes the kinetic energy of all the avalanche mass, it implicates trivially a further kinetic head reduction.

Turbulence Effect. The effect of the turbulence is modelled by a proportional kinetic head loss at each VALANCA step: $-\Delta KH = td \cdot KH$. This formula involves that a velocity limit is imposed “de facto”. A generic case with a maximum value of slope may be always transformed in the worst case of an endless channel with constant maximum value slope. In this case an asymptotic value of kinetic head is implied by infinite formula applications and, therefore, a velocity limit is deduced.

Avalanche Mass Outflows. Outflows computation is performed in two steps: determination of the outflows minimising the “height” differences in the neighbourhood [12] [13] and determination of the shift of the outflows.

The minimisation algorithm defines a central cell quantity d to be distributed, $d = \sum_{i=0}^n f[i]$ where $f[i]$ is the flow towards the cell i ($f[0]$ is the part of d , which remains in the central cell); $h[i]$, $0 \leq i \leq 6$ are the quantities that specify the “height” of the cells in the neighbourhood, to be minimised by contribution of flows: more precisely, the algorithm minimises the expression [16]:

$$\sum_{\{(i,j)|0 \leq i < j \leq 6\}} |(h[i] + f[i]) - (h[j] + f[j])| \quad (1)$$

Avalanches are rapid flows and imply a run up effect, depending on the kinetic head associated to debris flow. As a consequence, the height minimisation algorithm [17] [18] is applied, considering for the central cell $h[0] = A[0] + KH[0] + D[0]$ and the $d = TH[0]$; $h[i] = A[i] + TH[i] + D[i]$, $1 \leq i \leq 6$ for the adjacent cells; note that $KH[0]$ accounts for the ability of climbing a slope for the flowing avalanche. The minimisation algorithm determines the flows $f[i]$, $0 \leq i \leq 6$ toward the neighbouring cells ($f[0]$ is the part of d which is not distributed); such flows minimise the expression (1).

The mass centre co-ordinates x and y of moving quantities are the same of all the avalanche mass inside the cell and the form is ideally a “cylinder” tangent the next edge of the hexagonal cell (Fig.2). The height difference $h[0] + d - h[i]$ determines an ideal slope $\theta[i]$ between the two cells 0 and i ; a preliminary test is executed in order to account the friction effects, that prevent avalanche outflows, when $\tan \theta[i] < fc$. An ideal length “ l ” is considered between the avalanche mass centre

of central cell and the centre of the adjacent cell i including the slope $\theta[i]$, it represents the maximum allowed path of the outflow.

The $f[i]$ shift “ sh ” is computed for avalanche outflow according to the following simple formula, that averages the movement of all the mass as the mass centre movement of a body on a constant slope with a constant friction coefficient:

$$sh = v \cdot t + g \cdot (\sin\theta - fc \cdot \cos\theta) \cdot t^2 / 2, \text{ with “}g\text{” the gravity acceleration, the initial velocity } v = \sqrt{2g \cdot KH}.$$

The motion involves three possibilities: (1) only internal flow, the shifted cylinder is completely internal to the central cell; (2) only external flow, all the shifted cylinder is external to the central cell inside the adjacent cell; (3) the shifted cylinder is partially internal to the central cell, partially external to the central cell, the flow is divided between the central and the adjacent cell, forming two cylinders with mass centres corresponding to the mass centres of the internal flow and the external flow.

The kinetic head variation is computed according to the new position of internal and external flows, while the energy dissipation was considered as a turbulence effect in the previous elementary process.

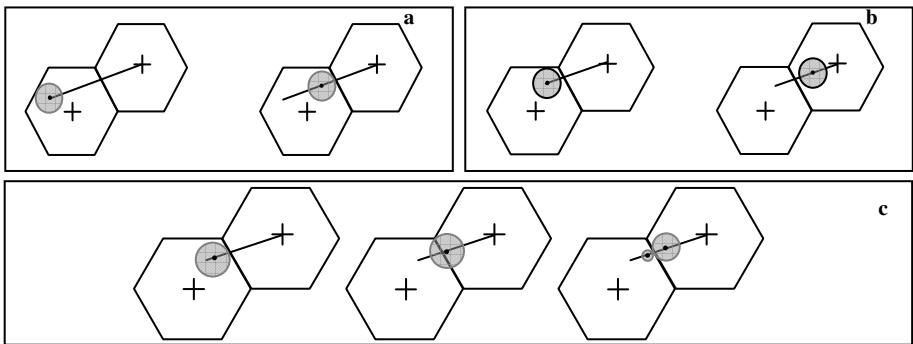


Fig. 2. Determination of the outflow shift. (a) All the cylinder remains in the cell: the flow is only internal and contributes to change the new centre mass of the cell. (b) All the cylinder leaves the cell: the flow is only external. (c) Part of the cylinder crosses the cell: there are both internal and external flows.

Flows Composition. When avalanche mass outflows are computed, the new situation involves that external flows leave the cell, internal flows remain in the cell with different co-ordinates and inflows (trivially derived by the values of external flows of neighbour cells) could exist. The new value of TH is given, considering the balance of inflows and outflows with the remaining snow mass in the cell. A kinetic energy reduction is considered by loss of flows, while an increase is given by inflows: the new value of the kinetic head is deduced from the computed kinetic energy. The co-ordinates determination is calculated as the average weight of X and Y considering the remaining snow mass in the central cell, the internal flows and the inflows.

2.2 Some Relevant Differences between VALANCA and ASCA

ASCA [10] is a CA model for the simulation of snow avalanches, with many analogies with VALANCA. As a matter of fact, both are two-dimensions models based on hexagonal cells and are ruled by the same flow distribution algorithm [11]. The models distinctions are synthesized in the following points.

ASCA shares with many CA models [12] an approach, that doesn't permit to make velocity explicit: a fluid amount moves from a cell to another one in a CA step, which corresponds usually to a constant time. This implies a constant local "velocity" in the CA context of discrete space/time, even if a kind of flow velocity emerges by averaging on the space (i.e. considering clusters of cells) or by averaging on the time (e.g. considering the average velocity of the advancing flow front in a sequence of CA steps). VALANCA, instead, inherits characteristics of the last releases of SCIDDICA ([15], [16]), that introduce coordinates of mass centre of flows and computes their shift.. In this case, velocity is locally explicit (cf. Section 2.1). The introduction of mass centers have introduced improvements in simulations in terms of fitness, despite a slight worsening in execution times over the considered simulations.

Note that energy losses related to the kinetic head (cf Section 2.1) are handled in ASCA according different formulae, deduced by approaches of PDE type.

In ASCA, the considered test-case snow avalanche was extremely rapid and thus characterised by relevant run-up effects, whose physical meaning is the minimum height of an obstacle needed to stop the motion of a mass with thickness moving at a certain velocity. Here, the run-up is determined by the thickness of the snow plus a fictitious height, which corresponds to the kinetic head and represents (Fig.1), in the minimization process, a conservative quantity which has to be distributed among the neighboring cells in order to reach the conditions of maximum stability. At the contrary, in the VALANCA model, the run-up effect for fast moving snow avalanches is expressed in a different manner. Here, the kinetic head is not considered as a whole with the snow (i.e., it is not considered as a mobile part during the minimization process) but computed separately from it in order to explicitly consider the physical characteristics related to energy loss and avalanche velocity related to the kinetic head itself (cf. Section 2.1).

3 VALANCA Applications to Real Cases of Snow Avalanches

VALANCA was developed in ANSI C++ in order to obtain both a well structured and extensible source code. The program is characterised by a command line interface that allows the user to interactively control all input/output and simulation, in order that the user can to visualize the simulation in real time. Through the viewer module, it is also possible to observe the DTM over which the phenomenon evolves and perform both a visual and quantitative comparison with the real case in terms of the fitness function f_a , later defined. A first validation and calibration of the parameters of VALANCA have been performed by back-simulating two snow avalanches in Davos (Switzerland) occurred in 2006 and well described in Errera [19]. The same set of

parameters have permitted to reproduce the two considered snow avalanches with a great level of accuracy.

First calibrations were performed as usual by a preliminary trial and error method; results were enough good that it was not necessary to use our automatic “long time” calibration techniques [20] (e.g. by means of Genetic Algorithms).

Simulation times depend on the number of active cells processed for each step and on the number of steps, necessary to complete the phenomenon: 10000 cells in a step last approximately 0.5 s for a 2.4GHz dual-core PC. Gotschnawang takes about three minutes with a 199 x 283 matrix excluding interactive graphical output.

During winter season the Davos area is affected by a big number of events. Furthermore, test avalanches were selected since they were well known in terms of areal path, thickness, deposit, velocity during the propagation etc.

The first event analysed (Gotschnawang) is quite challenging since it occurred in an open slope, while the second one (Rüchitobel) represents an interesting example of channelled snow avalanche. Detailed data of snow avalanches were available, among them the release area and volume, avalanche path, the spatial distribution and local thickness of the final deposit, the snow density in the snow cover and in the deposit. Furthermore, in a few cases the propagation velocities could be estimated at specific points. Snow cover entrainment occur in both cases, and also traces of fluidized flow were detected. However, the model simulates, as first attempt, only dense flows.

Both events were simulated by using a 5 m cell-size DTM of the area derived from aereophotogrammetry. Several simulations were performed in order to calibrate the parameters and best results are described in what follows.



Fig. 3. Gotschnawang avalanche. Outline of the 2006-01-20 Gotschnawang avalanche (Davos, Switzerland). The extent of the fracture line is indicated by the dotted line.

A first comparison between the real events and the simulated ones is performed by a fitness function f_a [20] concerning areas and computed by the following formula $\sqrt{(R \cap S)/(R \cup S)}$, where R is the set of cells affected by the avalanche in the real event and S the set of cells affected by the avalanche in the simulation. It returns a normalised value between 0 (complete failure) and 1 (perfect simulation). Further comparisons for good values of $f_a (>0.7)$ are performed on erosion and deposits.

3.1 The Gotschnawang Snow Avalanche

On January 20, 2006 a dry-snow avalanche was released artificially from the Gotschnawang slope in the Parsenn ski area (municipality of Klosters, eastern Switzerland) to protect the intermediate station of the Gotschna telepherique. The avalanche covered an area of 230 ha and it was 750 m long (projected length) with a vertical drop of 460 m. This amounts to a runout angle of 31.5° , which is a fairly large value for a dry-snow avalanche of this size. The site is an open but very hummocky slope with a straight track. The release area was 400 m wide, and we estimated the release depth at 40 cm (Fig.3).

The avalanche developed a fluidized layer whose deposits are visible both on the right-hand side and in the frontal part. The fluidized layer ran up to 50 m farther than the dense part and its mass was estimated at 100 tons, compared to 3000 tons for the dense part.

The snow avalanche was back-analysed by the VALANCA model by using a 5 m cells DTM by taking into account the release area, the portion of the slope covered by the snow mass and the erosion during the propagation.

A surprising areal agreement between the real event and the simulation was achieved (Fig.4) corresponding to a fitness $f_a = 0.92$.

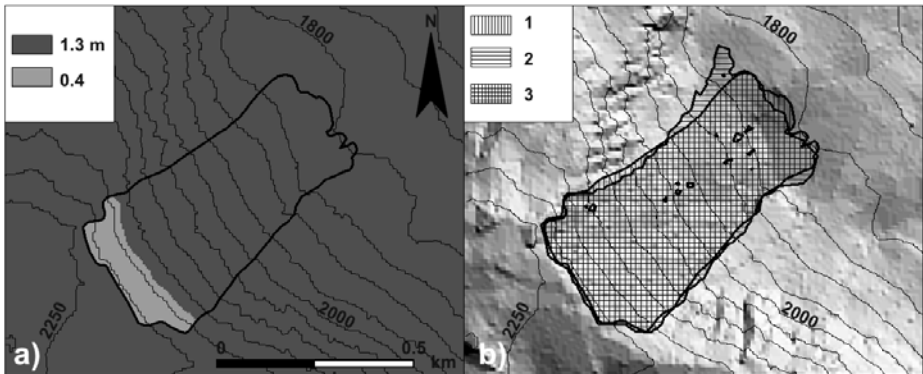


Fig. 4. The 2006 avalanche in Gotschnawang: (a) snow cover with detachment area and real event, (b) post-event DEM, superposition (3) of real (1) and simulated (2) event.

3.2 The Rüchitobel Snow Avalanche

On January 18, 2006 a dry-snow avalanche occurred in the Rüchitobel gully, Dischma Valley (Davos, eastern Switzerland) (Fig.5). The avalanche covered an area of 10 ha and it was 1167 m long (projected length) with a vertical drop of 630 m. The irregularly shaped release area was 195–290 m wide, with an estimated average release depth of 90 cm. For about 650 m (projected length), the flow was channelled in a winding gully whose bottom is around 10–15 m wide and was covered by 0.5–1.5 m

deep deposits from this avalanche, stacked over those from earlier ones, while the new snow was eroded completely.

On the upper parts of the gully banks and to the sides, the traces of fluidised layer were clearly observable to a height of 10–15 m above the gully bottom in that the snow was completely eroded away, without any deposits. In the most pronounced bend, the angle between the top flow marks of the fluidised part on either side and the tilt of the surface of the dense deposit indicated maximum flow velocities 28–38 m/s and 10–20 m/s, respectively.

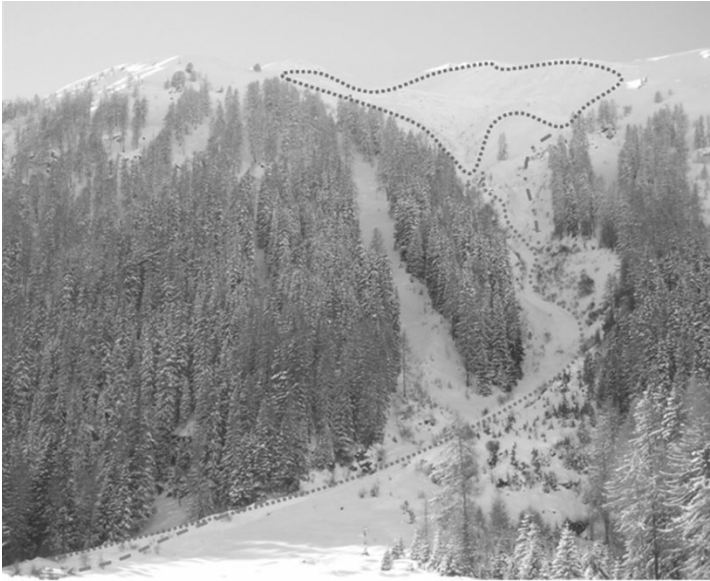


Fig. 5. Rüchitobel avalanche. Outline of the 2006-01-18 Rüchitobel avalanche (Davos, Switzerland). The extent of the fracture line is indicated by the top dotted line.

On the left-hand side and also at the distal end of the runout area, the deposit showed the characteristic features expected from fluidised flow. The mass of the dense deposit was estimated at 4000 *tons* while the fluidised one was only 40–50 *tons* (1% of the avalanche mass).

The snow avalanche was back-analysed by the VALANCA model by using a 5 m cells DTM by taking into account the release area, the portion of the slope covered by the snow mass and the erosion during the propagation. The best simulation results corresponded to a value of f_a close to 0.81 which is considered a satisfying preliminary result if the complex geometry of the avalanche path is taken into account (Fig.6).

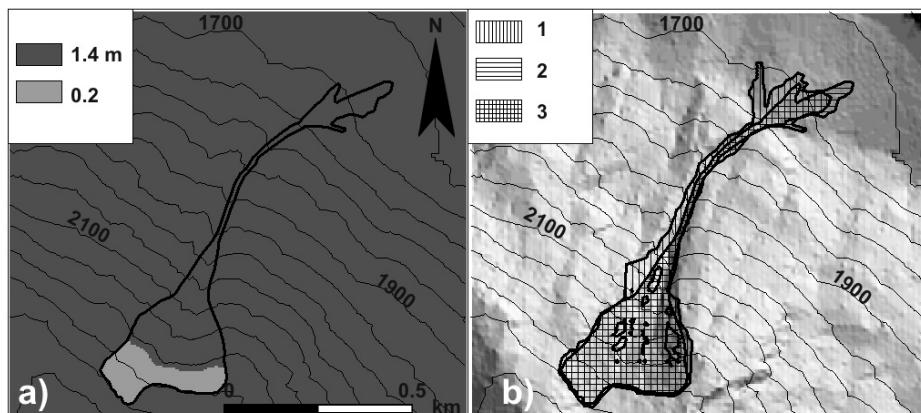


Fig. 6. The 2006 avalanche in Rüchitobel: (a) snow cover with detachment area and real event, (b) post-event DEM, superposition (3) of real (1) and simulated (2) event

4 Conclusions

The CA model VALANCA has been developed which is suitable for the simulation of snow avalanche dynamics. Preliminary validation and calibration of the model have been performed by back-analysing the Rüchitobel and Gotschnawang 2006 snow avalanches. Preliminary results, discussed in this paper, prove the ability of the model to simulate such a type of events in a satisfying way. The real path of the snow avalanche has been well simulated in both open and channelled slopes. However in spite of the encouraging results several improvements (mainly in the numerical management of the erosion and snow entrainment and in the avalanche velocity) are still needed in order to use such a model for forecasting analyses of snow avalanches propagation and their interaction with structures and human settlements.

Furthermore, we are confident that VALANCA could be usefully used in hazard analyses for snow avalanches. With this aim, applications to other cases of different type of snow avalanches have been already planned.

Acknowledgments. The authors wish to thank Dr Dieter Issler for useful comments and suggestions.

References

1. Issler, D., Gauer, P., Schaer, M., Keller, S.: Staublawineneignisse im Winter 1995: Seewis (GR), Adelboden (BE) und Col du Pillon (VD). SLF Internal Report 694. Eidg. Institut für Schnee- und Lawinenforschung, Davos, Switzerland (1996)
2. Schaerer, P.A., Salway, A.A.: Seismic and impact-pressure monitoring of flowing avalanches. *J. Glaciol.* 26(94), 179–187 (1980)
3. Issler, D., Errera, A., Priano, S., Gubler, H., Teufen, B., Krummenacher, B.: Inferences on flow mechanisms from snow avalanche deposits. *Annals of Glaciology* 49, 187–192 (2008)

4. Issler, D., Gauer, P.: Exploring the significance of the fluidised flow regime for avalanche hazard mapping. *Annals of Glaciology* 49, 193–198 (2008)
5. Harbitz, K.: A survey of computational models for snow avalanche motion. Tech. Rep. Fourth European Framework Programme (ENV4-CT96-0258) Avalanche Mapping, Model Validation and Warning Systems (1999)
6. Perla, R.I., Cheng, T.T., McClung, D.M.: A two parameter model of snow avalanche motion. *J. Glaciol.* 26(94), 197–202 (1980)
7. Eglit, M.E.: Mathematical and physical modelling of powder snow avalanches in Russia. *Annals of Glaciology* 26, 281–284 (1998)
8. Pudasaini, S.P., Hutter, K.: *Avalanche Dynamics: Dynamics of Rapid Flows of Dense Granular Avalanches*. Springer, Berlin (2007)
9. Ancey, C.: Snow Avalanches. In: Balmforth, N.J., Provenzale, A. (eds.) *Geomorphological Fluid Mechanics: Selected Topics in Geological and Geomorphological Fluid Mechanics*, pp. 319–338. Springer, New York (2001)
10. Barpi, F., Borri-Brunetto, M., Delli Veneri, L.: Cellular-Automata Model for Dense-Snow Avalanches. *Journal of Cold Regions Engineering* 21(4), 121–140 (2007)
11. Kronholm, K., Birkeland, K.W.: Integrating spatial patterns into a snow avalanche cellular automata model. *Geophysical. Research Letters* 32, L19504, 4pages (2005), doi:10.1029/2005GL024373
12. Di Gregorio, S., Serra, R.: An empirical method for modelling and simulating some complex macroscopic phenomena by cellular automata. *FGCS* 16, 259–271 (1999)
13. Avolio, M.V.: Esplicitazione della velocità per la modellizzazione e simulazione di flussi di superficie macroscopici con automi cellulari ed applicazioni alle colate di lava di tipo etneo. Ph. D. Thesis. Dept. of Mathematics, University of Calabria (2004) (in Italian)
14. D’Ambrosio, D., Di Gregorio, S., Gabriele, S., Gaudio, R.: A Cellular Automata Model for Soil Erosion by Water. *Phys. Chem. Earth (B)* 26(1), 33–39 (2001)
15. Avolio, M.V., Lupiano, V., Mazzanti, P., Di Gregorio, S.: Modelling combined subaerial-subaqueous flow-like landslides by Cellular Automata. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008. LNCS*, vol. 5191, pp. 329–336. Springer, Heidelberg (2008)
16. Avolio, M.V., Lupiano, V., Mazzanti, P., Di Gregorio, S.: A Cellular Automata Model for Flow-like Landslides with Numerical Simulations of Subaerial and Subaqueous Cases. In: Wohlgemuth, V., Page, B., Voigt, K. (eds.) *EnviroInfo 2009*, vol. 1, pp. 131–140. Shaker Verlag GmbH, Aachen (2009)
17. Avolio, M.V., Crisci, G.M., D’Ambrosio, D., Di Gregorio, S., Iovine, G., Rongo, R., Spataro, W.: An extended notion of cellular automata for surface flows modelling. *WSEAS Trans. Circuits Syst.* 2, 1080–1085 (2003)
18. D’Ambrosio, D., Di Gregorio, S., Iovine, G.: Simulating debris flows through a hexagonal cellular automata model: SCIDDICA S3-hex. *NHESS* 3, 545–559 (2003)
19. Errera, A.: Analisi dei processi fisici nelle valanghe di neve e conseguenze sulla pianificazione territoriale. Applicazione all’area di Davos (CH). Msc Thesis. Dept. of Earth Sciences, University of Milan-Bicocca (2007) (in Italian)
20. D’Ambrosio, D., Spataro, W.: Parallel evolutionary modelling of geological processes. *Parallel Computing* 33(3), 186–212 (2007)

Tracking Uncertainty in a Spatially Explicit Susceptible-Infected Epidemic Model

Jan M. Baetens* and Bernard De Baets

KERMIT, Department of Applied Mathematics, Biometrics and Process Control,
Ghent University, Coupure links 653, Gent, Belgium
{jan.baetens,bernard.debaets}@ugent.be

Abstract. In this paper we conceive an interval-valued continuous cellular automaton for describing the spatio-temporal dynamics of an epidemic, in which the magnitude of the initial outbreak and/or the epidemic properties are only imprecisely known. In contrast to well-established approaches that rely on probability distributions for keeping track of the uncertainty in spatio-temporal models, we resort to an interval representation of uncertainty. Such an approach lowers the amount of computing power that is needed to run model simulations, and reduces the need for data that are indispensable for constructing the probability distributions upon which other paradigms are based.

Keywords: continuous cellular automaton, epidemic spread, imprecision, uncertainty.

1 Introduction

As a consequence of their rigorous formulation of macroscopic phenomena, as well as their rich history which can be traced back to the development of modern calculus during the 17th and 18th century, and during which their efficacy has been proven manifold, ordinary differential equations (ODEs) are generally resorted to for describing (a)biological processes, as illustrated extensively in the work of Murray [16], whereas partial differential equations (PDEs) are mostly employed if one is not merely interested in the process' temporal dynamics but also in the spatial patterns it generates, such as the spread of an epidemic [17]. Further, in order to cope with the variability inherent to natural processes, researchers have resorted to stochastic DEs [20], fuzzy DEs [11,19], and to massive Monte Carlo (MC) simulations [26] in the hope that the simulation results obtained through a model based upon one of these approaches would agree to a larger extent with the described process than the outcome of their deterministic counterparts do. Yet, each of these paradigms suffers from a serious drawback. More specifically, stochastic DEs are difficult to solve analytically, or require advanced numerical techniques in order to find an approximate solution, the theory on fuzzy ODEs, and, especially fuzzy PDEs is still maturing, while much computing time and effort is needed to perform MC simulations.

* Corresponding author.

To overcome these barriers, we propose an interval-valued continuous cellular automaton (ICCA) for describing epidemic spread if there is imprecision involved about the magnitude of the initial outbreak or the epidemic’s characteristics. In essence, an ICCA can be regarded as a continuous CA (CCA) – also known as a coupled-map lattice – formulated by Kaneko [9], in which a cell’s state is represented by an interval in \mathbb{R} , and not longer by a single real value.

A short overview of the mathematical preliminaries that are essential for a clear understanding of this paper is given in Section 2. In the third section we introduce the ICCA that can be used to describe epidemic spread if there is imprecision involved in the magnitude of the initial outbreak or the epidemic’s characteristics. The former is addressed in the first part of the final section, while the latter is investigated more closely in the second part of this paper’s final section.

2 Preliminaries

For the sake of clarity we state the definition of an ICCA on an arbitrary tessellation of a 2-dimensional Euclidean space. This paradigm constitutes an extension to the CCA paradigm since the states of the spatial entities are represented by an interval-valued in \mathbb{R} , while it also entails an extension to the classical CA paradigm conceptualized by von Neumann [27] since it allows irregular tessellations of \mathbb{R}^2 .

Definition 1. (*Interval-valued continuous cellular automaton*)

An interval-valued continuous cellular automaton (ICCA) \mathcal{C} can be represented as a sextuple

$$\mathcal{C} = \langle \mathcal{T}, S, s, s_0, N, \Phi \rangle ,$$

where

- (i) \mathcal{T} is a countably infinite tessellation of a 2-dimensional Euclidean space \mathbb{R}^2 , consisting of cells $c_j, j \in \mathbb{N}$.
- (ii) S is an infinite set of intervals, where

$$S \subseteq [\mathbb{R}] = \{[y_1, y_2] \mid y_1 < y_2 \wedge y_1, y_2 \in \mathbb{R}\} .$$

- (iii) The output function $s : \mathcal{T} \times \mathbb{N} \rightarrow S$ yields the state value of cell c_j at the t -th discrete time step, i.e. $s(c_j, t) = [s_1(c_j, t), s_2(c_j, t)]$.
- (iv) The function $s_0 : \mathcal{T} \rightarrow S$ assigns to every cell c_j an initial state, i.e. $s(c_j, 0) = s_0(c_j)$.

- (v) The neighborhood function $N : \mathcal{T} \rightarrow \bigcup_{p=1}^{\infty} \mathcal{T}^p$ maps every cell c_j to a finite sequence $N(c_j) = (c_{j_k})_{k=1}^{|N(c_j)|}$, consisting of $|N(c_j)|$ distinct cells c_{j_k} .

- (vi) $\Phi = (\phi_j)_{j \in \mathbb{N}}$ is a family of functions

$$\phi_j : S^{|N(c_j)|} \rightarrow S ,$$

each ϕ_j governing the dynamics of cell c_j , i.e.

$$s(c_j, t + 1) = \phi_j(\tilde{s}(N(c_j), t)),$$

$$\text{where } \tilde{s}(N(c_j), t) = (s(c_{j_k}, t))_{k=1}^{|N(c_j)|}.$$

In the framework of this paper, we define N in such a way that $N(c_j)$ yields the Moore neighborhood of c_j , consisting of those cells $c_k \in \mathcal{T}$ that share either a vertex or a line segment with c_j . Sticking to this neighborhood function, it becomes straightforward to map \mathcal{T} on a undirected graph $G(V, E)$, with vertex set $V = \mathcal{T}$, while E represents the edge set of G , containing an edge between c_j and c_k if $c_k \in N(c_j)$. Furthermore, in the remainder of this paper we restrict to the family of ICCA for which ϕ_j is the same for all $c_j \in \mathcal{T}$.

Definition 2. (*Homogeneous interval-valued continuous cellular automaton*)

A homogeneous interval-valued continuous cellular automaton (ICCA) is an ICCA fulfilling premises (i)-(v) of Definition 1 and for which there exists a $\Theta : \bigcup_{k \in \mathbb{N}} S^k \rightarrow S$ such that

$$s(c_j, t + 1) = \Theta(\tilde{s}(N(c_j), t)).$$

Essentially, the construction of a homogeneous ICCA is less intricate than the composition its generalized counterpart given by Definition 1 since only one function Θ should be chosen that governs the dynamics of every $c_j \in \mathcal{T}$. Actually, most studied CA, such as rule 30 or the Game of Life [8], belong to this CA family.

3 A Spatially Explicit Model for Describing Epidemic Spread

3.1 The Model

The rich variety of CCA- and CA-based models that has been developed during the last decade for describing various spatial biological phenomena such as epidemics [6,14,28], population dynamics [3,5], tumor growth [13,23,24], biofilm development [21,22] and many other phenomena [10,25] is illustrative for the suitability of such models to mimic complex bioprocesses.

In a forthcoming work, Baetens and De Baets [2] propose a generalized CCA for modelling various biological processes that are traditionally described by means of PDEs. In this paper we focus on an epidemic sweeping through a geographical region, and which involves only non-reproducing susceptible and infected individuals. The spatio-temporal dynamics of such an epidemic can be captured by the following set of difference equations

$$\begin{cases} H(c_j, t + 1) = H(c_j, t) - H(c_j, t) \sum_{c_k \in N(c_j)} w_{jk} F(\mathbf{U}_j, d_{jk}) U(c_k, t) \\ U(c_j, t + 1) = U(c_j, t) + H(c_j, t) \sum_{c_k \in N(c_j)} w_{jk} F(\mathbf{U}_j, d_{jk}) U(c_k, t) \end{cases} \quad (1)$$

where $H(c_j, t)$, resp. $U(c_j, t)$, represent the fraction of susceptible (healthy), resp. infected (unhealthy) individuals within polygon c_j at the t -th time step such that $H(c_j, t) + U(c_j, t) = 1$, at all t , and for all c_j , F is a function describing the effect of landscape and connectivity characteristics, embodied in \mathbf{U}_j , on the epidemic, d_{jk} is the distance measured on a graph between the polygons c_j and c_k . Further, w_{jk} is a weighing factor, representing the influence of every $c_k \in N(c_j)$ in the determination of $H(c_j, t + 1)$. A brief analysis of Eq. (1) shows that it has two fixed points, namely $(H_j^*, U_j^*) = (0, 1)$ and $(H_j^*, U_j^*) = (1, 0)$. Clearly, this model may be regarded as a discrete analog of a PDE-based SI-model, such as described in [17].

Taking into account that $H(c_j, t) + U(c_j, t) = 1$, at all t , and for all c_j , we observe that the epidemic's dynamics can be tracked by considering only one of the system's equations. Further, by assuming that the region is spatially homogeneous, meaning that F does not depend on \mathbf{U}_j , we can reduce Eq. (1) to

$$U(c_j, t + 1) = U(c_j, t) + H(c_j, t) \sum_{c_k \in N(c_j)} w_{jk} H(d_{jk}) U(c_k, t), \quad (2)$$

where we introduced the function H , for which

$$H(d_{jk}) = \begin{cases} \nu_0, & \text{if } d_{jk} = 0, \\ \nu_1, & \text{if } d_{jk} = 1, \end{cases} \quad (3)$$

with ν_0 and ν_1 quantifications of the epidemic's virulence. These measures have to be chosen such that

$$\sum_{c_k \in N(c_j)} w_{jk} H(d_{jk}) \leq 1, \quad \forall c_j, \quad (4)$$

assuring that $0 \leq U(c_j, t) \leq 1$, at all t , and for all c_j . Finally, we put $w_{jk} = \frac{1}{8}$ for all j, k and $j \neq k$ and $w_{jk} = 1$ if $j = k$. Consequently, c_j 's eight nearest neighbours influence $U(c_j, t + 1)$ to the same degree.

3.2 Incorporating Uncertainty in the Proposed Model

Clearly, the above outlined model is deterministic since it yields exactly the same simulation result if its parameters and the initial condition from which it evolves are unchanged. In order to turn it into a stochastic model that is capable of grasping the variability inherent to natural process, commonly, it is presumed that $H(c_j, 0)$ and $U(c_j, 0)$, or the model's parameters follow a prescribed type of probability distribution, and the model is simulated through extensive MC simulations. Unfortunately, the latter require much computing time and effort, whereas a thorough construction of the aforementioned distributions demands a considerable amount of spatial data, which, mostly, cannot be collected easily.

For that reason, we propose to characterize uncertainty by using an interval representation of the variables and parameters in Eq. (2). More specifically, in the remainder of this paper $H(c_j, t)$ and $U(c_j, t)$ are considered intervals

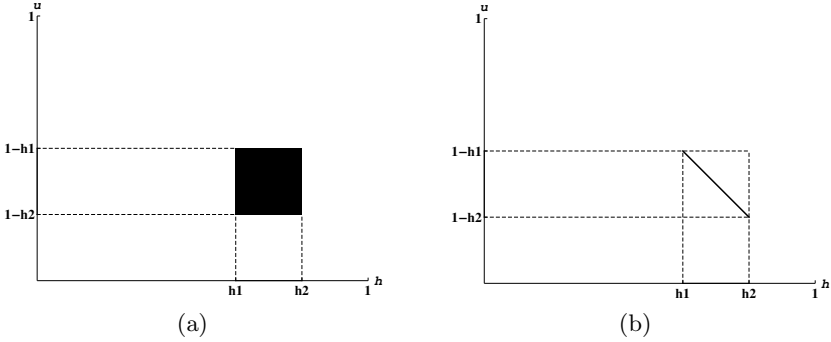


Fig. 1. Non-interactive (a) and interactive (b) intervals

in $[0, 1]$, such that we can write $H(c_j, t) = [h_1(c_j, t), h_2(c_j, t)]$ and $U(c_j, t) = [u_1(c_j, t), u_2(c_j, t)]$. Since the notion of uncertainty is for most researchers inextricably bound up with probability distributions, we will refer to an interval characterization of uncertainty as imprecision.

Seen the absence of derivatives or advanced mathematical functions in Eq. (2), it is relatively straightforward to evolve the system's spatio-temporal dynamics by means of basic interval arithmetic [15]. However, there is one pitfall that complicates the calculations, and inevitably leads to faulty conclusions if discarded. More precisely, one has to bear in mind the coupling between $H(c_j, t)$ and $U(c_j, t)$ through the condition $H(c_j, t) + U(c_j, t) = 1$, at all t , and for all c_j , which makes that $H(c_j, t)$ and $U(c_j, t)$ cannot take values independently of each other, so they can be termed interactive variables [7]. Hence, mathematical operators may not act on all couples in $H(c_j, t) \times U(c_j, t)$ (Fig. 1(a)), but only on the couples contained in

$$\{l(h_1(c_j, t), u_2(c_j, t)) + (1 - l)(h_2(c_j, t), u_1(c_j, t)) \mid l \in [0, 1]\}, \quad (5)$$

such as depicted in Figure 1(b).

In view of the existing interactivity, we can then write

$$H(c_j, t) + U(c_j, t) = [h_1(c_j, t) + u_2(c_j, t), h_2(c_j, t) + u_1(c_j, t)], \quad (6)$$

and, analogously,

$$H(c_j, t) \cdot U(c_j, t) = [\min(h_2(c_j, t) \cdot u_1(c_j, t), h_1(c_j, t) \cdot u_2(c_j, t)), \max(h_2(c_j, t) \cdot u_1(c_j, t), h_1(c_j, t) \cdot u_2(c_j, t))]. \quad (7)$$

4 Simulation Study

In this section two sources of imprecision in Eq. (2) are examined more closely. The first one concerns the initial condition $U(c_j, 0)$ that is necessary to iteratively solve Eq. (2), and which can be deduced from spatial epidemiological data

that are becoming increasingly available as indicated by Beale [4]. Nonetheless, we must be aware of the imprecision that can be present in the outbreak data, as illustrated only recently by the outbreak of H1N1 [12]. Analogously, the parameters in Eq. (3) might only be known imprecisely. This is regarded as the second source of imprecision. All simulations reported in this section were performed in Mathematica 7.0 (Wolfram Research, Inc.) on a desktop PC with an Intel Dual Quad Core 3.16 GHz processor. Although a square tessellation consisting of 101×101 polygons was used in this paper, the described simulations could easily be performed when an irregular tessellation is employed. Such an irregular tessellation seamlessly complies with the spatio-temporal data in vector format [1], which are commonly available through geographical information systems and can contribute considerably to a more accurate description of bioprocesses. No boundary conditions had to be imposed since we employed differentiated neighborhood structures along the tessellation's boundaries. As such, the use of periodic boundary conditions, which is rather questionable if one wants to simulate an epidemic over a given geographical extent, is avoided.

4.1 Imprecise Initial Conditions

Often only imprecise information is available on the magnitude of an epidemic during its initial stage. This kind of imprecision can be incorporated easily in the model (Eq. (2)), by choosing $U(c_m, 0)$ an interval in $[0, 1]$, where c_m represents the polygon in which the epidemic broke out. In practice, the choice of an appropriate interval should be based upon expert opinions, though, in order to exemplify the ability of the formerly described discrete modeling paradigm to incorporate imprecision it suffices to adopt an arbitrary initial condition such as

$$U(c_j, 0) = \begin{cases} [0.2, 0.4] & , \text{ if } j = m, \\ [0, 0] & , \text{ else.} \end{cases} \quad (8)$$

Further, we assume that reliable information is available on the virulence of an epidemic, which allows us to assess $\nu_0 = 0.5$ and $\nu_1 = 0.5$, meaning that the spread of an infection in a polygon c_j can be equally attributed to infected individuals living in c_j as to infected individuals residing in c_j 's neighborhood $N(c_j)$.

Figure 2 shows the center of $U(c_j, t)$ at two, five, ten and fifteen time steps after an epidemic outbreak occurred in the polygon c_m , as well as the length of the interval $U(c_j, t)$, denoted $|U(c_j, t)|$, at the same number of time steps. The former gives information on the expected proportion of infected individuals in every c_j , while the latter quantifies the imprecision that is related to this proportion. For reasons of clarity, we limited the depicted spatial extent of this figure to polygons through which the epidemic sweeps during the considered simulation period. This figure clearly shows that the imprecision originating from the imprecisely known proportion of infected individuals at $t = 0$ in the polygon c_m , propagates circularly like the epidemic wavefront. Since Figure 2(f) clearly shows that $|U(c_j, t)| \rightarrow 0$ as t increases, we may conclude that the ICCA

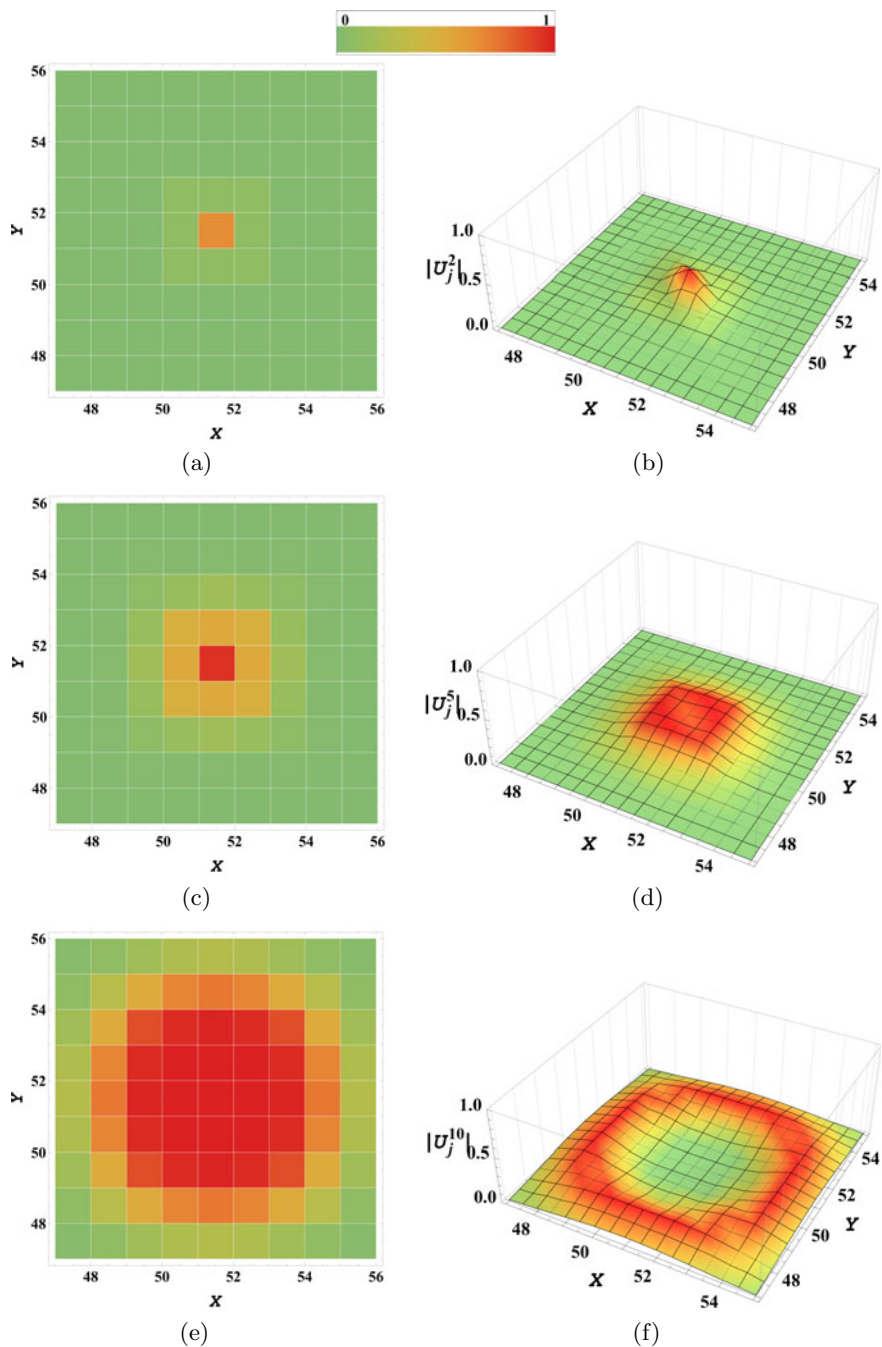


Fig. 2. Proportion of infected individuals, more precisely, the center of $U(c_j, t)$ (a,c,e), and the length of the interval $U(c_j, t)$, denoted $|U(c_j, t)|$ (b,d,f), two (a-b), five (c-d) and ten (e-f) time steps after an epidemic broke out in the center polygon c_m of a square tessellation with an initial magnitude given by Eq. (8)

evolves towards its fixed point $(H_j^*, U_j^*) = (0, 1)$, notwithstanding we imposed an imprecise initial condition. It should be stressed that this intuitive tendency would not have been observed if the formerly described interactivity between the model’s variables was discarded.

4.2 Imprecise Epidemic Properties

In this section we consider the model given by Eq. (2) with imprecise initial conditions given by Eq. (8), but in addition we assume that also ν_0 is only known imprecisely. The imprecision related to this parameter can be taken into account by representing it as an interval in $[0, 1]$. For that purpose, we choose $\nu_0 = [0.2, 0.5]$. Figure 3 visualizes the length of the interval $U(c_j, t)$, denoted $|U(c_j, t)|$ two, five and ten time steps after an epidemic struck c_m . Comparing Figs. 2(b), 2(d) and 2(f) on the one hand, and Fig. 3 on the other hand, one clearly sees that $U(c_j, t)$ is considerably larger when both the initial condition and ν_0 are only imprecisely known. We verified that the maximum attainable interval length spanned the entire unit interval in polygons more distant from c_m as the wavefront propagates, which can be attributed to the successive non-interactive multiplication of ν_0 and $U(c_j, t)$. Nevertheless, $U(c_j, t)$ tended to a crisp number as $t \rightarrow \infty$ since the ICCA evolves towards its fixed point $(0, 1)$.

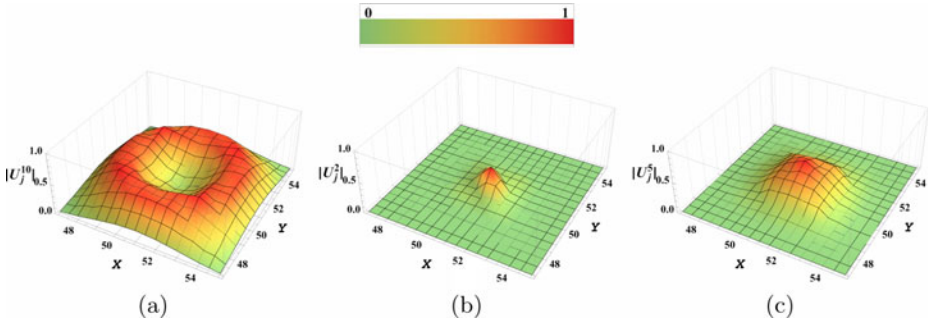


Fig. 3. Length of the interval $U(c_j, t)$ two (a), five (b) and ten (c) time steps after an epidemic broke out in the center polygon c_m of a square tessellation with an initial magnitude given by Eq. (8), and $\nu_0 = [0.2, 0.5]$

5 Discussion

Notwithstanding it was shown in the previous section that the proposed ICCA provides a means to deal with the imprecise nature of an epidemic in terms of the size of its initial magnitude and its properties, we must emphasize that the proposed paradigm is still to be improved in such a way that the quantities enclosed in a given interval are assigned a possibility with which they occur. Then, the impreciseness would no longer be represented by an interval that

merely encloses all possible values, but by a so-called fuzzy interval as depicted for illustration in Fig. 4. Unavoidably, this brings with it a complication of the calculations involved that then should be done in the light of Zadeh's [29] extension principle making that approach computationally less efficient than an ICCA. Naturally, an ICCA is trivially efficient since it merely requires two parallel model simulation, one for each interval limit, whereas a multiple of them would be required by MC methods. Of course, the additional computational effort enables to treat uncertainty in a much more informative way. Yet, by relying on fuzzy intervals one could combine an efficient numerical recipe, which would still not demand as many model simulations as needed for MC methods since Nguyen's [18] theorem can be invoked, with a model output bearing a much higher information degree.

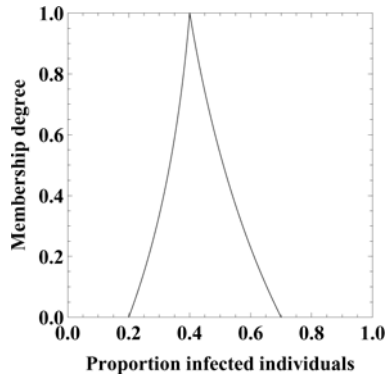


Fig. 4. An exemplary fuzzy interval in which every quantity in $[0, 1]$ is assigned a grade of membership to the set of infected individuals

6 Conclusions

In this paper we showed that imprecise information, described by means of intervals, can be used easily within a spatially explicit epidemic spread model that is based upon the continuous CA paradigm. More precisely, we demonstrated that uncertainty arising from both imprecise initial conditions or an epidemic's properties can be taken into account straightforwardly. The presented modeling framework is perfectly suited to cope with the growing importance and availability of spatio-temporal data. In forthcoming work, we will extend the presented model to cover also recovered individuals such that it can serve as a full-fledged alternative to PDE-based models. Besides, instead of representing imprecision by means of intervals, fuzzy numbers could be employed if there is information on the possibility with which every element in the interval occurs.

References

1. Baetens, J., De Baets, B.: Cellular automata on irregular tessellations. *Chaos, Solitons Fractals* (2010) (submitted)
2. Baetens, J., De Baets, B.: A generalized coupled-map lattice to model biological phenomena. *Mathematical Biology* (2010)(submitted)
3. Baltzer, H., Braun, P., Köhler, W.: Cellular automata models for vegetation dynamics. *Ecol. Modell.* 107, 113–125 (1998)
4. Beale, L., Abellan, J., Hodgson, S., Jarup, L.: Methodologic issues and approaches to spatial epidemiology. *Environ. Health Perspect.* 116, 1105–1110 (2008)
5. Dewdney, A.: Sharks and fish wage an ecological war on the toroidal planet. *Sci. Am.* 251, 14–22 (1984)
6. Doran, R., Laffan, S.: Simulating the spatial dynamics of foot and mouth disease outbreaks in feral pigs and livestock in Queensland, Australia, using a susceptible-infected-recovered cellular automata model. *Prev. Vet. Med.* 70, 133–152 (2005)
7. Dubois, D., Prade, H.: *Possibility Theory: an Approach to Computerized Processing of Uncertainty*. Plenum Press, New York (1988)
8. Gardner, M.: Mathematical games: The fantastic combinations of John Conway's new solitaire game 'Life'. *Scientific American* 223, 120–123 (1971)
9. Kaneko, K. (ed.): *Theory and Applications of Coupled Map Lattices*. John Wiley & Sons Ltd., Chichester (1993)
10. Kier, L., Seybold, P., Cheng, C.: *Modelling Chemical Systems using Cellular Automata*. Springer, Dordrecht (2005)
11. Lakshmikantham, V., Mohapatra, R.: Theory of fuzzy differential equations and inclusions. In: Agarwal, R., O'Regan, D. (eds.). *Series in Mathematical Analysis and Applications*, vol. 6. Taylor & Francis, New York (2003)
12. Lipsitch, M., Riley, S., Cauchemez, S., Ghani, A.C., Ferguson, N.M.: Ferguson: Managing and reducing uncertainty in an emerging influenza pandemic. *N. Engl. J. Med.* 361, 112–115 (2009)
13. Mallet, D., De Pillis, L.: A cellular automata model of tumor-immune system interactions. *J. Theor. Biol.* 239, 334–350 (2006)
14. Milne, J., Fu, S.: Epidemic modelling using cellular automata. In: *Proc. ACAL 2003*, Canberra, pp. 43–57 (December 2003)
15. Moore, R.: *Interval Analysis*. Prentice Hall, Englewood Cliffs (1966)
16. Murray, J. (ed.): *Mathematical Biology: I. An Introduction*, 2nd edn. Springer, Berlin (1993)
17. Murray, J. (ed.): *Mathematical Biology: II. Spatial Models and Biomedical Applications*, 3rd edn. Springer, Berlin (2007)
18. Nguyen, H.: A note on the extension principle for fuzzy sets. *J. Math. Anal. Appl.* 64, 369–380 (1978)
19. Oberguggenberger, M., Pittschmann, S.: Differential equations with fuzzy parameters. *Math. Comput. Modell. Dyn. Syst.* 5, 181–202 (1999)
20. Øksendal, B.: *Stochastic Differential Equations: An Introduction with Applications*. Springer, Berlin (2003)
21. Picioreanu, C., van Loosdrecht, M., Heijnen, J.: Mathematical modeling of biofilm structure with a hybrid differential-discrete cellular automaton approach. *Biotechnol. and Bioeng.* 58, 101–116 (1998)
22. Pizarro, G., Griffeath, D., Noguera, D.: Quantitative cellular automaton model for biofilms. *J. Environ. Eng.* 127, 782–789 (2001)

23. Preziosi, L.: *Cancer Modelling and Simulation*. Chapman & Hall, Boca Raton (2003)
24. Qi, A., Zheng, X., Du, C., An, B.: A cellular automaton model of cancerous growth. *J. Theor. Biol.* 161, 1–12 (1993)
25. Schiff, J.: *Cellular Automata: A Discrete View of the World*. John Wiley & Sons Ltd., Chichester (2008)
26. Ulam, S.: The monte carlo method. *J. Am. Stat. Ass.* 44, 335–341 (1949)
27. von Neumann, J., Burks, A.: *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign (1966)
28. White, S., del Rey, A., Sanchez, G.: Modeling epidemics using cellular automata. *Appl. Math. Comput.* 186, 193–202 (2007)
29. Zadeh, L.: Fuzzy sets. *Inf. Control* 8, 338–353 (1965)

A Proximal Space Approach for Embedding Urban Geography into CA Models

Ivan Blečić, Arnaldo Cecchini, and Giuseppe A. Trunfio

Department of Architecture and Planning - University of Sassari,
Palazzo Pou Salit, Piazza Duomo 6, 07041 Alghero, Italy
{ivan,cecchini,trunfio}@uniss.it
<http://www.lampnet.org>

Abstract. In the great majority of urban models based on Cellular Automata (CA), the concept of proximity is assumed to reflect two fundamental sources of spatial interaction: (1) the accessibility of places and (2) the distance “as the crow flies”. While the geographical space defined by the latter clearly has an Euclidean representation, the former, based on the accessibility, does not admit such a regular representation. Very little operational efforts have been undertaken in CA-based urban modelling to investigate and provide a more coherent and cogent treatment of such irregular geometries, which indeed are essential and crucial feature of urban geography. In this paper, we suggest an operational approach – entirely based on cellular automata techniques – to model the complex topology of proximities arising from urban geography, and to entangle such proximity topology with a CA model of spatial interactions.

Keywords: urban cellular automata, land-use dynamics, proximal space, irregular neighbourhood, informational signal propagation, informational field.

1 Introduction

The idea of spatial interaction in CAs is strongly related to that of proximity. Indeed, a CA transition rule is always, by definition, a function describing the relation between a cell and its neighbouring, *viz.* proximal cells. And the nature of proximity depends fundamentally on the kind of geometry we are using to describe the underlying geographical space.

When modelling urban dynamics based on spatial interactions, we are always implicitly or explicitly making assumptions on the nature of proximities within an urban geography, and are hence seeking for its suitable geometrical representation.

On that account, we can start by saying that in the great majority of CA urban models, the “proximities” are assumed to reflect two fundamental sources of spatial interaction: (1) the accessibility and (2) the distance “as the crow flies”. The geographical space defined by the latter kind of proximity has clearly an Euclidean representation, and thus the proximity may be defined in a form of a *regular* spatial distance, for example as a special case of the Minkowski distance.

However, in the case of the former kind of proximity, the one based on the so called accessibility, the situation is profoundly different. The accessibility refers here to a

measure of how (how much, how easy, how quickly) places are mutually accessible (i.e. reachable) from one another to human beings. Such accessibility may therefore be further subdivided by the means of transportation (pedestrian, bicycle, automobile, heavy vehicle, railways and so on) as they all give rise to different accessibilities of places. Anyhow, in all these cases, the accessibility itself is deeply determined by the relevant underlying urban geography. For example, the web of pedestrian, road, railways and underground transportation networks substantially shape such geography, bringing about a highly *irregular* geometry of the accessibility-type proximity. This type of proximity manifestly does not admit any possibility of a regular representation, let alone Euclidean. Indeed, an Euclidean representation of the geometry of the accessibility-type would be to a large extent inappropriate and fundamentally flawed.

Considering these rather straightforward observations, it is remarkably surprising that very little operational efforts (e.g. [1]) have been undertaken in CA-based urban modelling to investigate and provide a more coherent and cogent treatment of such irregular geometries, which indeed are essential and crucial feature of urban geography.

The lack of treatment of this feature is for instance easily seen in two families of urban CA models which in derived, extended, specialised or inspired-by forms have been often used for CA-based urban simulation: the so called Constrained Cellular Automata (CCA) [2-4] and those based on the SLEUTH approach [5-7]. In both these two families of models, the CA does not adopt a strictly local neighbourhoods, and therefore does indeed simulate spatial interactions over greater distances, but the distance is intended exclusively in the Euclidean as-the-crow-flies sense. The same general approach is described in the attempts to comprehensively present and discuss the theory and application of urban CA (see for example [8,9]).

Being such landscape of CA applications to urban phenomena as it is, there have been, to be fair, invitations from theoretical standpoints to develop a more appropriate understanding of the concept of nearness, to give it a deeper geographical meaning, in a way, to entail it with a thick geographical theory. Indeed, such a line of reasoning may almost directly be derived from the notion of *proximal* space, coming from the research in ‘cellular geography’ [10] which set the basis for the so called *geo-algebra* approach proposed by Takeyama and Coucleis [11]. In this latter paper, the homogeneity of cells’ neighbourhoods has been questioned precisely on the ground that every cell may have different neighbourhood defined by relations of “nearness” between spatial entities, where “nearness” can mean both topological relation or generic “behavioural” (e.g. functional) influence.

In this paper, we take on the task to suggest an operational approach – entirely based on cellular automata techniques – to model the complex topology of proximities arising from urban geography, and to entangle these and such proximity topology with a CA model of spatial interactions.

2 Proximal Spaces as Informational Fields

The approach we take to describe the irregular geometry of the accessibility-type proximity is to assume that each cell emits an informational signal propagating throughout

the cellular space. However, the signal propagation is not uniform, but depends on the “propagation medium” which the signal encounters. This means that, starting from the emitting cell, the signal is diffused in all directions, but the decay of the signal’s intensity depends on the state (e.g. land use) of the cells crossed by the signal. As a consequence of this informational signals emission, each cell generates an *informational field* around itself, whose shape and intensity at every cell of the cellular space depends on the states of the cells along all the paths the signal propagates.

To see how these general concepts may relate and be applied to urban context, we can for example think of a model in which the above described signals propagates better (i.e. with a lower rate of decay) along the roads, and that they easily spill over to the cells surrounding the roads. Another example could be a railway transportation network. Here, the signal would propagate smoothly along the railway, but would not by model design be allowed to spill over to the surrounding cells, except starting from the cells corresponding to railway stations.

To sum up, the beforehand suggested method allows us to generate an irregular geography-based “informational field” around each cell. In other words, seen from another point of view, every cell receives a set of signals of different type and intensity from other (potentially every other) cells. Once the informational fields are generated, the CA transition rules ought to be stated in a way to combine the received signals as the input information.

The hereby suggested strategy of modelling proximities by the means of information signals propagating through cellular space is similar in spirit to the “at-a-distance interaction fields” proposed in [12]. The specific contribution of our proposal should therefore be seen in its attempt to apply and embed these concepts into *urban* CAs and to conceive a particular operational simulation approach for that purpose.

3 An Application to a CA Model

The experimental setting to demonstrate and discuss the above ideas was a 2D CA composed of square cells providing a raster representation of a geographical area. The state of every cell represents its land-use type, which can be of one of the following eight types: *residential*, *industrial*, *commercial*, *agriculture*, *road*, *railways*, *railway station*, *public services/facilities*. The latter four types are considered as static and thus cannot change nor be transformed endogenously during the simulation. Starting from a given initial configuration, the automaton evolves in discrete steps simulating the land-use dynamics of the area.

At each simulation step, the execution of the CA model is divided into two distinct phases: (1) *informational fields generation phase* and (2) *land-use dynamics phase*.

3.1 Informational Fields Generation Phase

This phase of the CA execution has the task to generate the informational fields (of the kind described in section 2) around each cell. Specifically, at the first step of this phase, the cells having residential, industrial, commercial, or public services land use

are made to emit an informational signal σ . Each signal holds and carries the following information: (1) the ID of the *source* cell, (2) the source cell's *land use*, (3) the *propagation rule*, and (4) the signal's intensity $\bar{\sigma}$. During the subsequent steps, every signal held by a cell is transmitted to its Moore-neighbouring cells, provided that the signal's intensity is above a predefined threshold.

Signals are subject to a decay of intensity defined by their propagation rule. In general, a propagation rule is expressed as a function of the land uses of both the sender and the receiver cell. The functioning of the propagation rules is therefore grounded on a land-use "in-out matrix". More specifically, this matrix defines the coefficients of decay of the informational signal on the basis of the land use combination of the signal's outgoing and incoming cells. An example of such an in-out matrix is shown in Table 1. The decay coefficients in this table reflect the observations on the accessibility signal propagation exemplified above in section 2.

Table 1. Example of land-use "in-out matrix" of coefficients used for calculating the decay of a signal propagating from an outgoing to an incoming cell

Incoming cell	R	C	I	A	PS	Ro	Rw	RwS
Outgoing cell								
R (Residential)	0.60	0.60	0.60	0.10	0.60	0.90	0.00	0.95
C (Commercial)	0.60	0.60	0.60	0.10	0.60	0.90	0.00	0.95
I (Industrial)	0.60	0.60	0.60	0.10	0.60	0.90	0.00	0.95
A (Agricultural)	0.10	0.10	0.10	0.10	0.10	0.90	0.00	0.95
PS (Pub. services)	0.60	0.60	0.60	0.10	0.60	0.90	0.00	0.95
Ro (Road)	0.90	0.90	0.90	0.90	0.80	0.95	0.00	0.95
Rw (Railway)	0.00	0.00	0.00	0.00	0.00	0.00	0.98	0.95
RwS (Railway station)	0.95	0.95	0.95	0.90	0.95	0.95	0.98	0.95

To account for two relevant modes of spatial interaction discussed in the introduction (see section 1), two types of propagation rules are defined in the model:

- Regular, Euclidean-space propagation, by which the signal decay is a function of the Euclidean distance from the signal's source (and therefore does not depend on the land uses crossed by the signal);
- Irregular, Proximal-geography-space propagation, by which the decay of the signal's intensity differs depending on the land uses of the cells being crossed by the signal.

As an example, consider Fig. 1 where the intensity of the fields originated by some emitting cells is depicted in an urban-like environment characterised by the presence of a network of roads. In case (a) the signal decay does not account for the current cells' land uses: this type of propagation rule makes signals able to inform the receiving cell about the existence of the source cell and also about the level of their spatial distance in the Euclidean sense. In case (b) the signals' intensity propagates with smaller decay along the roads and railways: this propagation rule allows a receiving cell for being informed about the existence and the accessibility of the source cell.

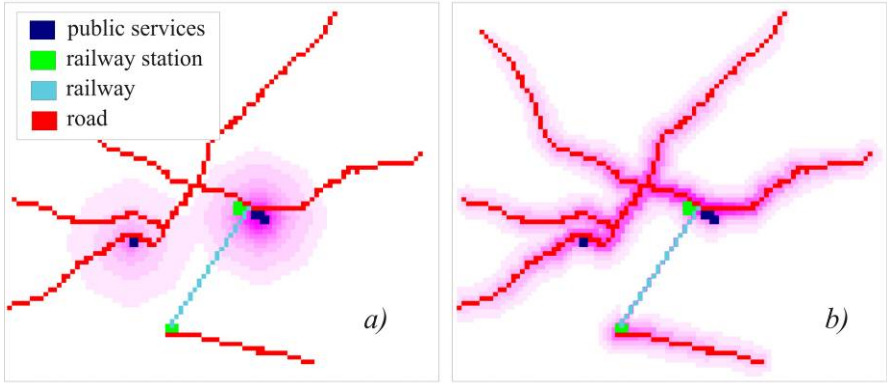


Fig. 1. Comparison between the diffusion of two type of signals originating from the same cells. In case (a) signals decay does not account for the current cells' land uses; in case (b) signals' intensity propagate preferentially along cells representing roads and railways.

The informational fields generation phase ends when during a time step eventually no signal propagates further throughout the CA. This condition is satisfied when every cell has already sent all its signals and all the received signals are of the intensity below the predefined propagation threshold.

3.2 Land-Use Dynamics Phase

At the end of each informational fields generation phase, every cell holds a set Σ of signals σ which are used as the input information for the subsequent land-use dynamics phase.

This phase is based on the computation of the so-called *transition potentials* $P_j \in [0, 1]$ expressing the propensity of the land to acquire the j -th land use. In [2-4], where the hereby employed concepts of transitional potentials have been developed, the cell neighbourhood is a circular region of a given radius around the cell. Therefore, we adapted the thereby presented rules to our circumstances of irregular neighbourhood patterns as drawn by informational fields. Hopefully, we succeeded in maintaining the spirit of the spatial interaction principles inherent in the original rules.

Adapting and somewhat simplifying from [2], the transition potentials of every cell in our model are computed as:

$$P_j = \begin{cases} S_j Z_j N_j & \text{if } N_j \geq 0 \\ (1 - S_j Z_j) N_j & \text{if } N_j < 0 \end{cases} \quad (1)$$

where:

- $S_j \in [0, 1]$ is the cell *physical suitability* taking into account, for each land use j , features like slope or terrain aspect.
- Z_j is a Boolean value defining the exclusion of the j -th land use (for example due to zoning regulations or physical constraints)
- $N_j \in \mathbb{R}$ is the so called *neighbourhood effect*.

The latter represents the sum of all the relevant attractive and repulsive effects of land uses and land covers on the j -th land use which the current cell may assume. In the present model, and critically differently than in [2-4], N_j is computed using all the informational signals Σ received by the cell:

$$N_j = I_k \delta_{jk} + \sum_{\sigma \in \Sigma} f_{ij}(\sigma) \quad (2)$$

where:

- i denotes the type of the signal σ ;
- $f_{ij}(\sigma) \in [-1, 1]$ is a function giving the influence of a signal σ if type i on the use j which the cell may assume;
- δ_{ik} is 1 if $j = k$ and 0 if $j \neq k$, where k denotes the current land use of the cell for which the transition potential is under evaluation;
- the term $I_k \in [0, 1]$ accounts for the effect of the cell on itself (zero-distance effect) and represents an inertia due to the costs of transformation from one land use to another

Functions $f_{ij}(\sigma)$, accounting for different effects of a received signal on all the potential land uses, are assumed in the following form:

$$f_{ij}(\sigma) = a_{ij} \frac{1 - e^{-s_{ij} \bar{\sigma}}}{1 + e^{-s_{ij} \bar{\sigma}}} \quad (3)$$

where:

- $\bar{\sigma}$ denotes the intensity of the signal σ of type i ;
- $a_{ij} \in [-1, 1]$ is a parameter representing the maximum influence (positive or negative) of a signal σ of type i on the use j ;
- $s_{ij} \in [0, 1]$ is a parameter defining the sensitivity of the use j on a signal of type i ;

Fig. 2 shows some examples of functions $f_{ij}(\sigma)$ used in the model. It is important to note that, in spite of the simplicity of functions $f_{ij}(\sigma)$, the combination of all contributions given by Eq. (2), together with the different ways in which signals can propagate throughout the automaton, are able to effectively describe a variety of relevant situations. Consider for example the cell \mathbf{c}_a close to a road represented in Fig. 3 and suppose that, according to its current land use, it emits two signals, namely an Euclidean nearness signal σ_1 and an accessibility signal σ_2 (see also Fig. 1). Also consider a cell \mathbf{c}_b along the road, at a distance d from \mathbf{c}_a , receiving the two signals σ_1 and σ_2 emitted by \mathbf{c}_a and suppose that σ_1 has a repulsive effect on a potential land use j of \mathbf{c}_b (i.e. $a_{ij} < 0$ in Eq. 3) while σ_2 has an attractive effect on the same use j (i.e. $a_{ij} > 0$ in Eq. 3). The combination of the two effects, according to the model above described, leads to the transition potential contribution represented in Fig. 3 as a function of the distance d , which is characterised by an optimum distance (i.e. the position in which the contribution of \mathbf{c}_a to the potential towards the use j of \mathbf{c}_b is maximum) and a decay of the influence as d increases.

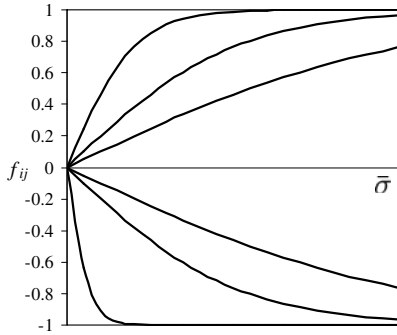


Fig. 2. Example graphs of the contribution to the transition potential given the signal intensity

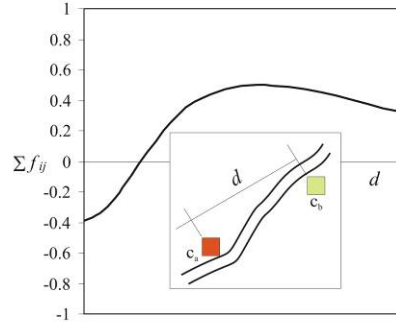


Fig. 3. Total effect produced by the cell c_a on different cells c_b as a function of their distance d

When all transition potential are computed, the following transition rule is applied to all cells of the automaton:

Transition rule: a cell of the land use k is transformed into the land use j , if P_j is the highest of all the cell's transitions potentials, and provided that $P_j > \lambda$ and $P_j - P_k > \epsilon$.

λ is a minimum threshold and ϵ is a minimum difference threshold for a cell to be transformed into another land use. The threshold ϵ incorporates the correction for age of the cell's land use k , namely the younger in terms of CA steps is the current land use, the greater is the threshold ϵ .

The rationale behind this transition rule is straightforward. A cell has a transition potential to transform into every possible land use. Of all the possible land uses, it will transform into the one having the strongest transition potential, provided that it is strong in absolute terms (greater than λ) and that it is strong enough to overturn the current land-use k (therefore it must hold $P_j - P_k > \epsilon$). This latter threshold (ϵ) accounts therefore for the inherent inertia or sunk costs of urban transformations. The correction for age of the threshold ϵ puts further consideration onto this inertia, by imposing that the required differential of transition potentials is greater had the cell just recently transformed into its current land use k . Subsequently, the threshold requirement ϵ is lowered gradually as the age of the current land use grows.

4 Example Runs

In this section we show, through a preliminary simulation exercise, some typical effects of taking into account the accessibility-type proximity in a CA-based land use simulation.

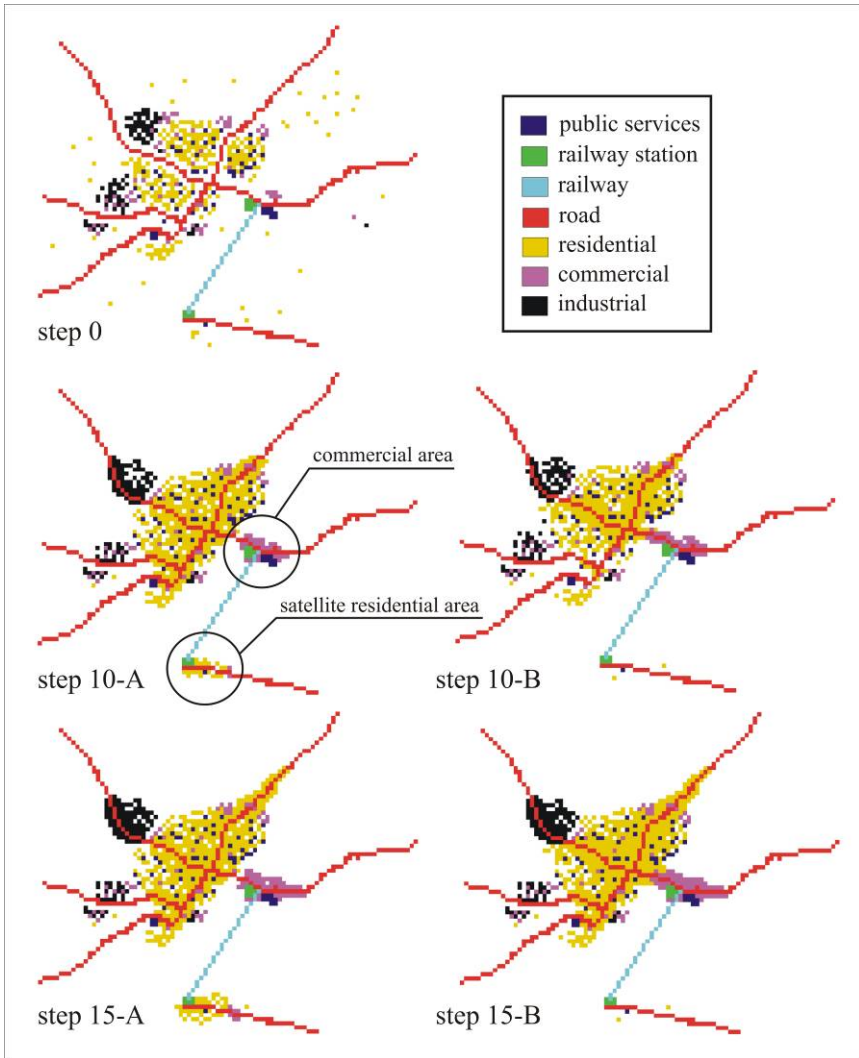


Fig. 4. Comparison between the outcomes of two runs of the model with different models of the proximity (left: accessibility-type, right: as-the-crow-flies type)

In particular, two simulations were executed, using the model described in section 3 and assuming as initial configuration the map of a hypothetical city (“step 0” in Fig. 4.)

In the first simulation (labelled “A” in Fig. 4), the model included the computation of the relevant accessibility fields (e.g. signals of the accessibility of public services or commercial areas) while in the second run (labelled “B” in Fig. 4) only signal propagating according to their Euclidean distances were admitted. Clearly, in the second case each cell of the automaton was not informed about accessibilities, being only aware of both the existence and the distance of all the relevant land uses (roads and railways included, in the spirit of [2-4]) within a radius given by the level of isotropic decay of the signals (see Fig. 1-a).

Each simulation was executed for 15 land-use transformation steps which, including the information fields generation phases, corresponded to 321 CA steps for simulation “A” and 169 CA steps for simulation “B”.

In Fig. 4, the output maps obtained for steps 10 and 15 are depicted. In particular, as the city evolves according to the model rules, the growth of a commercial area (see the highlighting circle) can be observed at step 10 in both simulations. At the same land-use step, the map produced by simulation “A” exhibits the development of a satellite residential area in a zone (see the highlighting circle) which is relatively distant from the newly developed areas but well connected by the railway. In simulation “B” the same satellite zone, although served by the railway, remains undeveloped (i.e. in case of model “B” the cells are not informed about the existence of highly accessible public services and commercial areas of new development at the other end of the railway connection).

Comparing the outcomes at step 15, we can observe that both the commercial area and the satellite residential area had further development, but the latter continued to be completely missed by simulation B.

In spite of its simplicity, the results of the above presented preliminary example indicate that including an improved description of the proximities arising in an urban geography can lead to different, and possibly more realistic, urban patterns.

5 Conclusions

In this paper, our primary aim was to suggest a possible modelling of the notion of proximity and proximal space arising in urban geography, and to employ it in urban CA. The point of departure was the idea that the proximity usually held to be relevant for spatial interactions cannot be assumed to exist in and as a regular Euclidean space, since the “distortions” of urban geography bring about highly irregular and complex topology of proximities. To account for this complexity, we have thereafter suggested a description of the proximal space as a set of informational signals propagating through the space, hence generating informational fields around each cell. Every such field exhibits different strength (intensity) at different points (i.e. cells) in space, since the irregularity of its geometry is inherently dependant on the different “propagation media” (road, railways, residential, commercial or any other area) crossed by the informational signals.

A cell of the automaton is then able to “know” its proximity to another by knowing the intensity of the signal – carrying on with the wave metaphor, shall we say “the radiation”? – it receives from that cell. Finally, the combination of all the received radiations is the information input to the CA transition rules through which to model the land-use dynamics.

In the example presented, our focus was not so much on the plausibility and theoretical foundation of the transition rules and of the proximity-based land-use urban dynamics. Rather, it was a quite expedient (and probably somewhat coarse) exemplification of how even existing models based on assumptions of spatial interaction may in a reasonably convenient manner be adapted to employ our more generalised and more geography-grounded notion and description of the proximal space.

References

1. Batty, M.: Distance in space syntax. CASA Working Papers (80). Centre for Advanced Spatial Analysis (UCL), London, UK (2004)
2. White, R., Engelen, G.: Cellular automata and fractal urban form: A cellular modeling approach to the evolution of urban land-use patterns. *Environment and Planning*, 1175–1199 (1993)
3. White, R., Engelen, G., Uljee, I.: The use of constrained cellular automata for high-resolution modelling of urban land use dynamics. *Environment and Planning B* 24, 323–343 (1997)
4. White, R., Engelen, G.: High-resolution integrated modelling of the spatial dynamics of urban and regional systems. *Computers, Environment and Urban Systems* 28(24), 383–400 (2000)
5. Clarke, K., Hoppen, S., Gaydos, L.: A self-modifying cellular automaton model of historical urbanization in the san francisco bay area. *Environment and Planning B* 24, 247–261 (1997)
6. Clarke, K.C., Gaydos, L.J.: Loose-coupling a cellular automaton model and GIS: long-term urban growth predictions for San Francisco and Baltimore. *International Journal of Geographic Information Science*, 699–714 (1998)
7. Project Gigalopolis, NCGIA (2003),
<http://www.ncgia.ucsb.edu/projects/gig/>
8. Benenson, I., Torrens, P.M.: Geosimulation: object-based modeling of urban phenomena. *Computers, Environment and Urban Systems* 28(1-2), 1–8 (2004)
9. Torrens, P.M., Benenson, I.: Geographic Automata Systems. *International Journal of Geographical Information Science* 19(4), 385–412 (2005)
10. Tobler, W.: Cellular geography. In: Gale, S., Olsson, G. (eds.) *Philosophy in Geography*, pp. 379–386. Reidel, Dordrecht (1979)
11. Takeyama, M., Couclelis, H.: Map dynamics: integrating cellular automata and GIS through Geo-Algebra. *Intern. Journ. of Geogr. Inf. Science* 11, 73–91 (1997)
12. Bandini, S., Mauri, G., Vizzari, G.: Supporting Action-at-a-distance in Situated Cellular Agents. *Fundam. Inform.* 69(3), 251–271 (2006)

Bone Remodelling: A Complex Automata-Based Model Running in BIOSHAPE

Diletta Cacciagrano, Flavio Corradini, and Emanuela Merelli

School of Science and Technology, University of Camerino, 62032, Camerino, Italy
`{name.surname}@unicam.it`

Abstract. Bone remodelling, as many biological phenomena, is inherently multi-scale, i.e. it is characterised by interactions involving different scales at the same time. At this aim, we exploit the *Complex Automata* paradigm and the BIOSHAPE 3D spatial simulator respectively (i) for describing the bone remodelling process in terms of a 2-scale aggregation of *uniform Cellular Automata* coupled by a *well-established* composition pattern, and (ii) for executing them in a *uniform* and integrated way in terms of shapes equipped with perception and movement capabilities.

On the one hand, the proposed model confirms the high expressiveness degree of Complex Automata to describe multi-scale phenomena. On the other hand, the possibility of executing such a model in BIOSHAPE highlights the existence of a general mapping - from Complex Automata into the BIOSHAPE native modelling paradigm - also enforced by the fact that both approaches result to be suitable for handling different scales in a uniform way, for including spatial information and for bypassing inter-scale homogenization problems.

1 Introduction

Nowadays, it is possible to observe biological systems in great detail: with a light microscope one can distinguish the compartments of a human cell, and with an electron microscope one can even see very small details such as proteins. At the same time, models for describing and simulating biological systems have comparable resolution regimes and work on different spatial and temporal scales: in the microscopic approach, molecular dynamics and Monte Carlo methods describe systems at the level of atoms or proteins while, in the macroscopic regime, continuum-based simulations model complete biological assemblies (but do not describe any explicit molecular information). Actually, a characteristic of biological complexity is the *intimate connection* that exists between different length scales. For instance, subtle changes in molecular structure as a consequence of a single gene mutation can lead to catastrophic failure at the organ level, such as heart failure from re-entrant arrhythmias that lead to ventricular fibrillation. But information flows equally in the reverse direction: mechanoreceptors at the cell level sense the mechanical load on the musculoskeletal system and influence gene expression via signal transduction pathways.

1.1 A Case Study: The Bone Remodelling Process

Old bone is continuously replaced by new tissue. This ensures that the mechanical integrity of the bone is maintained, but it causes no global changes in morphology: Frost defined this as *remodelling* [1]. Such a phenomenon can be considered “multi-scale” (see Fig. 1) since macroscopic behaviour and microstructure strongly influence each other.

Bone remodelling at tissutal scale. Two macroscopically different bone tissue types are distinguished: the *cortical* one - which is a rather dense tissue although it is penetrated by blood vessels through a network of canaliculi - and the *trabecular* one - which is porous and primarily found near joint surfaces, at the end of long bones and within vertebrae.

On a macroscopic level, remodelling might be regulated by mechanical loading, allowing bone to adapt its structure in response to the mechanical demands. It is well-known that trabeculae tend

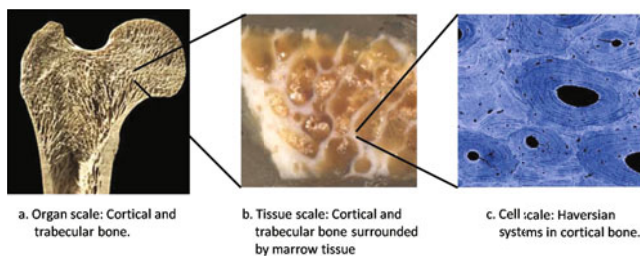


Fig. 1. Multiscale view of a human femur

to align with maximum stresses in many bones and greatly increase their load-carrying capacity without increasing mass, thus improving structural efficiency; mechanical stress also improves bone strength by influencing collagen alignment as new bone is being formed. Cortical bone tissue located in regions subject to predominantly tensile stresses has a higher percentage of collagen fibers aligned along the bone long axis. In regions of predominant compressive stresses, fibers are more likely to be aligned transverse to the long axis.

Bone remodelling at cellular scale[1]. Two main kinds of cells, namely *osteoclasts* (O_c) and *osteoblasts* (O_b), closely collaborate in the remodelling process in what is called a *Basic Multicellular Unit* (BMU). The organization of the BMUs in cortical and trabecular bone differs, but these differences are mainly morphological rather than biological.

The remodelling process begins at a quiescent bone surface (either cortical or trabecular) with the appearance of O_c s, which attach to the bone tissue matrix, form a ruffled border, create an isolated microenvironment, acidify it and dissolve the organic and inorganic matrices of the bone.

Briefly after this resorptive process stops, O_b s appear at the same surface site, deposit osteoid and mineralize it. Some O_b s are encapsulated in the osteoid

¹ For a more detailed description, see <http://courses.washington.edu/bonephys/physremod.html>.

matrix and differentiate to *osteocytes* (O_y). Remaining O_{bs} continue to synthesize bone until they eventually stop and transform to quiescent *lining cells* (L_c) that completely cover the newly formed bone surface and connect with the O_y s in the bone matrix through a network of canaliculi.

1.2 Motivations and Contribution of the Paper

The need of a multi-scale modelling approach. Bone remodelling was always subject of extensive studies in many fields of research: much of this research is based on reduction - i.e. isolating the various components to unravel their individual behaviour - without taking into account how mechanical forces are translated to structural adaptation of the internal cellular architecture [2,3,4,5], while other approaches relate density changes in bone directly to local strain magnitudes, abstracting from the underlying cellular processes (i.e. morphology and metabolic activity) [6,7,8,9].

Being bone remodelling an inherently multi-scale process, it is reasonable to bet on multi-scale modelling approaches [10,11,12], i.e. modelling approaches linking phenomena, models and information between various scales. To support this conjecture, it suffices to consider that the actual knowledge about this biological process shows several gaps *at different resolution degrees*:

- (*Tissue level*) There are some questions as to whether the orientation of collagen fibers in bone occurs through functional adaptation as the bone is being remodelled or is under genetic influence during development.
- (*Cell level*) BMU existence indicates that a coupling mechanism must exist between formation and resorption (i.e. among O_{bs} s and O_{cs} s). However the nature of this coupling mechanism is not known.
- (*Cell-Tissue level*) It is not so clear how mechanical forces can be expressed in cell activity and whether they are enough to explain remodelling. The current concept is that the bone architecture is also controlled by local regulators and hormones (mainly insulin-like growth factors, cytokines interleukin-1, interleukin-6 and RANKL) and that both local mechanical and metabolic signals are detected from O_y s. Whether this is true remains to be proven.

Homogenization and uniformity. Indeed, a multi-scale model is not necessarily more “faithful” than a single-scale one only because it is multi-scale. It is well-known that a multi-scale model can be more or less “faithful” according to what “single-scale” models are taken into account (for each scale) and how they are “homogenized” (i.e. integrated). Homogenization is in fact a very delicate and complex task - when “single-scale” models are heterogeneous, as well as when the biological systems to model admit different homogenization techniques - which can lead to loss of information between scales. As a consequence, a high uniformity degree among “single-scale” components implies the possibility of defining well-established homogenization rules and increasing the “faithfulness” of a multi-scale model in the whole.

Space and geometry. It is also well-known that the possibility of expressing spatial information is another important element which can add “faithfulness” to a biological model (not only multi-scale).

Consider, for instance, the microtubules: not only they have a specific geometry, but their polarity arises from the geometry of their tubulin components. Cytoplasm (of even the simplest cell) and enzymes are another excellent examples. The first contains many distinct compartments, each with its own specific protein set; even within a single compartment, localization of molecules can be influenced in many different ways, such as by anchoring to structures like the plasma membrane or the cytoskeleton. The latter, acting in the same pathway, are often found co-localised; as the product of one reaction is the substrate for the next reaction along the pathway, this co-localisation increases substrate availability and concomitantly enhances catalytic activity, by giving rise to increased local concentration of substrates.

Contribution of the paper. On the basis of the above observations, we exploit at the same time *Complex Automata* (CxA) [13] paradigm and BIOSHAPE² [14] 3D spatial simulator: the first for defining cellular and tissutal scale of bone remodelling as *uniform Cellular Automata* (CA) and aggregating them by a well-established composition pattern (see Section 2), while the latter for simulating, in a *uniform* and integrated way, both CAs in terms of shapes, equipped with perception and movement capabilities (see Section 3).

In particular, we deliberately approximate the biological process taking into account only mechanical stimuli and ignoring metabolic ones (see Subsection 2.1). This approximation does not deeply influence the tissutal scale, where the associated CA only models a lattice of BMUs; on the contrary, it is quite evident at cellular scale, where each single BMU is in turn described as a CA of O_y s, avoiding an explicit local regulator and hormone representation. If, on the one hand, the assumed approximation could influence the multi-scale model “faithfulness” w.r.t. the real phenomenon, on the other hand it does not influence the validity of the proposed crossing approach (modelling in CxA and simulating in BIOSHAPE) and the underlying mapping, being both CxA and BIOSHAPE able to describe and handle spatial lattices.

Although CxA paradigm was already equipped with its own execution environment [15] and BIOSHAPE with its native modelling language [16], the proposed crossing approach - here tested for a specific biological process - aims just to be the first step to state an expressiveness study between CxA and BIOSHAPE (see Section 4). In fact, the possibility of modelling bone remodelling as a CxA and executing it in a shape-based fashion in BIOSHAPE aims to highlight the existence of a *more general* encoding from CxA into BIOSHAPE. This conjecture is enforced by the fact that both approaches (i) are based on uniform single-scale models, (ii) rely on a Lattice Boltzmann Model-like time step scheme, (iii) can express spatial information and (iv) can bypass inter-scale homogenization problems.

² <http://cosy.cs.unicam.it/bioshape>

2 The Complex Automata Modelling Paradigm

The Complex Automata (CxA) [13] paradigm has been recently introduced for modelling multi-scale systems and, in particular, the process of development of stenosis in a stented coronary artery [17].

CxA building blocks are single-scale *Cellular Automata* (CA) (i) representing processes operating on different spatio-temporal scales, (ii) characterized by a uniform Lattice Boltzmann Model-like (LBM) update rule - and, as a consequence, execution flow (see Fig. 2 (b)) - (iii) mutually interacting across the scales by well-defined composition patterns³ (see Fig. 3).

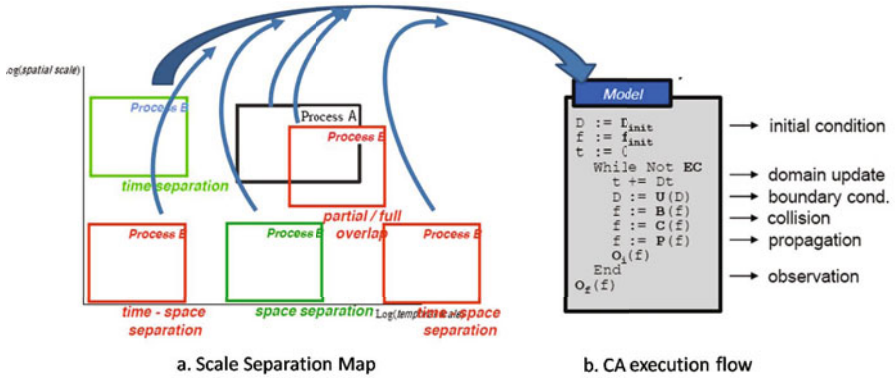


Fig. 2. a) Scale Separation Map; b) CA execution flow.

	Time Overlap		Time Separation	
Space Overlap	Single Domain Coupling through collision operator. <i>Snow transport, diffusion/ advection, ...</i>	Multi Domain Coupling through boundary condition. <i>Fluid structure, grid refinement, ...</i>	Single Domain Coupling through collision operator. <i>Forest-Savannah-Fire interactions</i>	Multi Domain Coupling through boundary, initial conditions. <i>Coral Growth, ...</i>
	Space Separation	Single Domain Coupling through collision operator. <i>Algae-Water ecological model, ...</i>	Multi Domain Coupling through boundary condition. <i>Wave propagation in two media, ...</i>	 Hierarchical Coupling Coupling through collision operator and initialization. <i>Suspension Fluid, ...</i>

Fig. 3. SSM and Composition patterns

³ Due to the lack of space, composition patterns are not discussed here and we refer to [18] for further details.

More in detail, the update rule of any CA is uniformly defined as a composition of three operators: *boundary condition* $B[\cdot]$ and *collision* $C[\cdot]$, both depending on external parameters, and *propagation* P , depending on the topology of the domain. Given a CA, (i) the B operator is needed to specify the values of the variable that are defined by its external environment (in the case of a LBM fluid simulation, the missing density distributions at the wall), (ii) the C operator represents the state update for each cell, and (iii) the P operator sends the local states of each cell to the neighbors that need it, assuming an underlying topology of interconnection.

Being the update rule of any CA uniformly defined, such composition patterns only depend on the CA spatio-temporal “positions” in a *Scale Separation Map* (SSM), where each CA is represented as an area according to its spatial and temporal scales (see Fig. 2 (a)). Formally:

Definition 1. A CxA \mathcal{A} is a graph (V, E) , where V - the set of vertices - and E - the set of edges - are defined as follows:

- $V = \{C_k \stackrel{\text{def}}{=} \langle (\Delta x_k, \Delta t_k, X_k, T_k), \mathbb{S}_k, \Phi_k, s_k^0, u_k \rangle \mid C_k \text{ is a CA} \}$ where $\forall C_k \in V$,
- $(\Delta x_k, \Delta t_k, X_k, T_k)$ denotes the spatio-temporal domain of C_k , i.e. Δx_k is the cell spatial size, X_k is the space region size, Δt_k is the time step and T_k is the end of the simulated time interval of C_k ;
- \mathbb{S}_k denote the set of states;
- $s_k^0 \in \mathbb{S}_k$ is the initial state;
- u_k is a field collecting the external data of C_k ;
- Φ_k is the update rule encoded in LBM style as follows

$$s_k^{n_k + \Delta t_k} = P \circ C[u_k^C][s_k^{n_k}] \circ B[u_k^B]$$

where $s_k^{n_k}, s_k^{n_k + \Delta t_k} \in \mathbb{S}_k$ denote resp. the state of C_k obtained as the numerical solution at the n_k -th time step and the one at the $(n_k + 1)$ -th time step.

- $E = \{E_{hk} \mid E_{hk} \text{ is a composition pattern between } C_h \text{ and } C_k\}$.

Finally, the numerical outcome of each C_k is denoted by $s^{T_k} \in \mathbb{S}_k$.

2.1 Multi-scale Trabecular Bone Remodelling in CxA

Assuming that O_y s act as mechano-sensors, the model - for simplicity proposed in 2D - consists of a CA, whose cells are in turn CAs: the “macro” CA (denoted by C_1) models a portion of trabecular bone as a lattice of BMUs (macroscopic slow process), while each “micro” CA (denoted by $C_{(i,2)}$, where i corresponds to the cell i in C_1) models a single BMU as a lattice of O_y s and their surrounding mineralized tissue (microscopic fast process).

A similar grid-based micro-macro spatial decoupling can be found in [19], where a discrete, agent-based stochastic model for studying the behavior of limb

bud precartilaginous mesenchymal cells in vitro is proposed. The model employs a multiscale spatial organization for cells and molecules as a two-dimensional discrete pixel grid. The coarsest resolution spatial scale (the cellular level) is the base spatial scale, and the molecular one is an integer ratio size of that base grid. Each molecule is considered to have a spatial extent of just one pixel, and each type of molecule has its own spatial grid independent of the other molecule types, so any number of molecule types can be defined, each with their own scale relative to the base spatial scale.

In our case, the “macro” cell size is linearly determined from the “micro” cell one, which is in turn derived from the $O_{y,s}$ estimated density in bone. Assuming that a cubic millimeter of fully mineralized tissue contains 16000 $O_{y,s}$, then a 3D lattice representing this unit volume should contain 25 ($\approx 16000^{1/3}$) cells in each side. Therefore, a cubic millimeter of bone could be modeled as a 3D lattice of 25^3 cells, matching with the data reported in [20]. As a consequence, a 2D cell lattice with a thickness of $1/25$ mm can be structured in 25^2 cells, matching also with the data presented in [21].

The “macro” neighborhood layout can be defined either as the simplest 2D Von Neumann neighborhood (4 cells) or as the 2D Moore one (8 cells), depending on how “local” we consider the remodelling process on a trabecular region (i.e., in other terms, how “local” we consider the propagation of remodelling activation state among BMUs). The “micro” neighborhood layout can be defined as the 2D Moore neighborhood.

Micro execution flow. The state of each cell j in $C_{(i,2)}$ at a time $t_{(i,2)}$ is defined by its mass fraction $m_{(i,2)}^j(t_{(i,2)})$, varying from 0 (bone marrow) to 1 (fully mineralized). The mechanical stimulus $F_{(i,2)}^j(t_{(i,2)}) = U_{(i,2)}^j(t_{(i,2)})/m_{(i,2)}^j(t_{(i,2)})$ - being $U_{(i,2)}^j(t_{(i,2)})$ the strain energy density of j at time $t_{(i,2)}$ - is calculated by the *Meshless Cell Method* [12] (MCM). Each cell j modifies its mass according to the error signal $e_{(i,2)}^j(t_{(i,2)})$ between the mechanical stimulus and the internal equilibrium state, determined by the condition $e_{(i,2)}^j(t_{(i,2)}) = 0$; when this condition does not hold, a local collision formula⁴ modifies the mass fraction ($m_{(i,2)}^j(t_{(i,2)}) + \Delta t_{(i,2)}$) to restore the equilibrium condition. Consequently, the change in mass modifies the stress/strain field in the bone and, therefore, the stimulus operating on j . This process continues until the error signal is zero or no possible mass change can be made. The convergence is satisfied when the change in density is small: if there is no convergence, the process continues with a new MCM analysis.

Macro execution flow. Similarly to the micro execution flow, the state of each cell i in C_1 is defined by the apparent density $m_1^i(t_1)$, which can vary from 0 (void) to 1 (fully mineralized tissue). An homogeneous apparent density distribution for any i corresponds to an isotropic material, while intermediate values represent trabecular architecture.

⁴ The formula can be selected from the approaches presented in [22].

A global MCM analysis evaluates the stress field $F_1^i(t_1)$ on i at a time t_1 , so defining the loading conditions operating on each i . We know that i modifies the microstructure by processes of formation/resorption (corresponding to $s^{T(i,2)}$, see below); this process results in formation and adaptation of trabeculae. Hence, the global MCM analysis is performed over the resulting structure to update the stress field until there is no change in the relative densities and there is no change in the stress field.

Micro-Macro composition pattern. Each $C_{(i,2)}$ is linked to C_1 by the “micro-macro” composition pattern, defined in Fig. 3 and maximized in Fig. 4. More in detail, C_1 takes input from explicit simulations of $C_{(i,2)}$ on each lattice site i at each time step Δt_1 , while each $C_{(i,2)}$ runs to completion, assuming that all $C_{(i,2)}$ are much faster than the macroscopic process and therefore are in quasi-equilibrium on the macroscopic time scale.

A close inspection of this coupling template shows indeed that upon each C_1 's iteration each $C_{(i,2)}$ executes a complete simulation, taking input from C_1 . In turn, each $C_{(i,2)}$'s output ($s^{T(i,2)}$) is fed into the C_1 's collision operator.

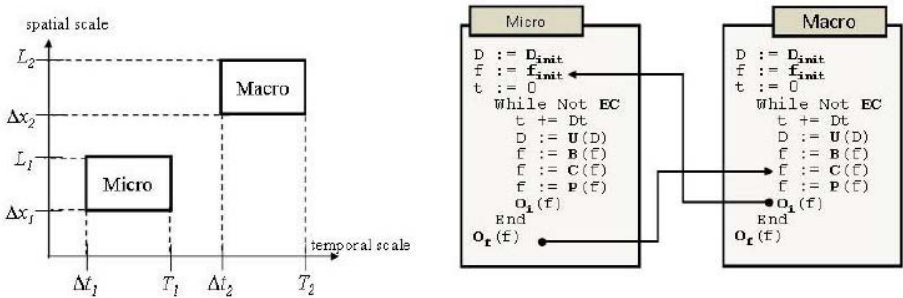


Fig. 4. Micro-Macro composition pattern

3 The BIOSHAPE Modelling and Simulation Environment

BIOSHAPE⁵ is a spatial 3D simulator which has been engineered in the perspective to be a *uniform, particle-based, space-oriented* multi-scale modelling and execution environment.

BIOSHAPE’s modelling approach treats biological entities of any size as geometric *shapes*, equipped with perception, interaction and movement capabilities. The behaviour of every shape, i.e. the way it interacts with other shapes and with the environment, is formally defined through a process algebra (namely, the *Shape Calculus* [16]). Every entity has associated its physical movement law: this approach guarantees granularity in entities management as everyone is treated uniformly but independently from the other ones.

Scale-independence property just follows by the uniform way biological entities of any size are treated. As a consequence, homogenization between single scales

⁵ A complete description of BIOSHAPE can be found in [14].

simply reduces to mappings between homogeneous representations at different granularity (i.e. zoom resolution) of the same biological system.

Time is simulated as a sequence of fixed time intervals called *time frames*; specific events - namely *perceptions* and *collisions* - occur during any time frame according to a well-established LBM-like sequence.

3.1 Multi-scale Trabecular Bone Remodelling in BIOSHAPE

BIOSHAPE can import and execute the CxA model described in Section 2 without substantial arrangements, since “micro” and “macro” CAs (i) can be encoded using a small subset of primitive (namely shapes, perception- and collision-driven interaction, communication and internal calculus), being simple spatial lattices, as well as (ii) can be easily homogenized, being homogeneous representations of the same process at different zoom resolution (micro-macro). Moreover, the BIOSHAPE software architecture has been already engineered from the perspective of supporting massive parallel computations⁶, a computational approach which is intrinsic to the CA paradigm and, as an immediate consequence, to the CxA one.

The CxA model of bone remodelling can be also executed in BIOSHAPE to provide a graphical simulation of the phenomenon at tissutal level. More in detail, the whole 2D trabecular tissue body can be visualized as a grid of square shapes in the fully mineralised part of the bone and in the fully fluid part (resp. full/green squares and void/black squares in Fig. 5 (B)). The bone *surface* can be represented also by squares, but decomposed using, as usual in visual graphics, five basic shapes able to “discretize” the trabecular surface (green shapes in Fig. 5 (C)).

The basic surface shapes are square, rectangle (1:2 aspect ratio), truncated square, two right angled triangles (side ratio of 1:1 and 1:2), and a trapezium glued with a rectangle and a triangle. They are grouped into 6 element families. Allowing for rotations and mirror images of these groups, only 29 stiffness matrices need to be defined, thus we can always find a good

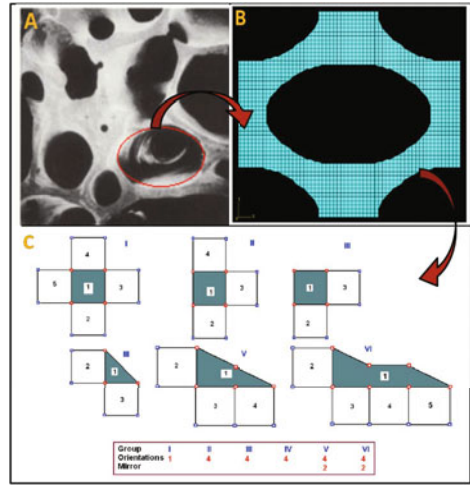


Fig. 5. Trabeculae (A). 2D grid (B). Surface shapes (C).

⁶ The current version is based on the UNICAM agent-based Java framework Hermes [23], a middleware supporting distributed applications and mobile computing. Currently, a porting on a Multiple Instruction Multiple Data (MIMD) architecture with message passing is under development.

representation of the border avoiding the need of a re-shaping primitive, which is not yet available in the current version of the simulator.

Each void/full/surface shape in the cell i has an associated mineralisation density. As a consequence, its dynamics is determined by the CxA model execution: in particular, a variation from $s_1^{n_1}$ to $s_1^{n_1+\Delta t_1}$ involving i determines the replacement of the shape in i with another void/full/surface shape - that one associated with the new density value.

4 Conclusion and Future Work

The CxA paradigm has been here taken into account to propose a uniform multi-scale model for the bone remodelling process. Such a model has been then imported and executed in BIOSHAPE.

Scale-independence property, ability of expressing spatial information and LBM-like time frame scheme are altogether elements which heavily draw up both modelling approaches. However, if on the one hand the CxA here proposed for bone remodelling confirms the high degree of expressiveness and flexibility of the CxA paradigm, on the other hand the possibility in BIOSHAPE to associate to each shape *its own* physical movement law (which can be different from that one associated to a neighbour) could make BIOSHAPE more expressive than a CxA. As a consequence, a formal investigation on the expressive power of the above modelling approaches seems reasonable.

We are also exploiting BIOSHAPE alone for defining a multi-scale model for bone remodelling where local regulators, O_b s, O_c s and their relative precursors, namely *pre-osteoblasts* (P_b) and *pre-osteoclasts* (P_c), are explicitly modelled as particles at cellular scale. The implementation of such a model in BIOSHAPE is quite natural, as it involves shapes that either move possibly attracted by biochemical signals or stand still. Also the composition of O_c s from P_c s is a primitive supported by the simulator. This is a promising approach, already exploited in [24], where basic simulation algorithms of the Celada-Seiden model for the immune response process are presented in terms of operations on abstract particle types, and where new algorithms for diffusion, proliferation and cell-cell interaction are defined as discrete versions of established continuous models.

We plan to tune and validate the new particle-based cellular model taking into account experimental data as well as those produced by the CxA-based model here proposed and by some available continuum-based descriptions [2,3,4,5,6,7,8,9]. We also plan to realize such tuning and validation procedures in the opposite direction.

Our believe is that particle-based tissutal and cellular views of bone remodelling turn to be helpful (i) to better understand the blurry synergy between mechanical and metabolic factors triggering bone remodelling, both in qualitative and in quantitative terms, and (ii) to develop a coherent theory for the phenomenon as modulated by mechanical forces and metabolic factors in a uniform way.

References

1. Frost, H.: Skeletal structural adaptations to mechanical usage (SATMU): 2. Re-defining Wolff's Law: the remodeling problem. *Anat. Rec.* 226, 414–422 (1990)
2. Boyle, W., Simonet, W., Lacey, D.: Osteoclast differentiation and activation. *Nature* 423, 337–342 (2003)
3. Harada, S., Rodan, G.: Control of osteoblast function and regulation of bone mass. *Nature* 423, 349–355 (2003)
4. Rouhi, G., Epstein, M., Sudak, L., Herzog, W.: Modeling bone resorption using mixture theory with chemical reactions. *Journal of Mechanics of Materials and Structures* 2(6) (2007)
5. Nauman, E., Satcher, R., Keaveny, T., Halloran, B., Bikle, D.: Osteoblasts respond to pulsatile fluid flow with short-term increases in pge (2) but no change in mineralization. *Journal of Applied Physiology* 90, 1849–1854 (2001)
6. Turner, C.: Toward a mathematical description of bone biology: the principle of cellular accommodation. *Calcified Tissue International* 65(6) (1999)
7. Cowin, C., Hegedus, D.: Bone remodeling i: A theory of adaptive elasticity. *Journal of Elasticity* 6, 313–326 (1976)
8. Cowin, C., Hegedus, D.: Bone remodeling ii: small strain adaptive elasticity. *Journal of Elasticity* 6, 337–352 (1976)
9. Huiskes, R., Ruimerman, R., Lenthe, G.V., Janssen, J.: Effects of mechanical forces on maintenance and adaptation of form in trabecular bone. *Nature* 405, 704–706 (2000)
10. Hunter, P.J., Li, W.W., McCulloch, A.D., Noble, D.: Multiscale modeling: Physio-me project standards, tools, and databases. *Computer* 39, 48–54 (2006)
11. Sloot, P.M.A., Hoekstra, A.G.: Multi-scale modelling in computational biomedicine. *Brief. Bioinform.* (2009)
12. Cristofolini, L., Taddei, F., Baleani, M., Baruffaldi, F., Stea, S., Viceconti, M.: Multiscale investigation of the functional properties of the human femur. *Philos. Transact. A Math. Phys. Eng. Sci.* 366(1879), 3319–3341 (2008)
13. Hoekstra, A.G., Falcone, J.L., Caiazzo, A., Chopard, B.: Multi-scale modeling with cellular automata: The complex automata approach. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 192–199. Springer, Heidelberg (2008)
14. Buti, F., Cacciagrano, D., Corradini, F., Merelli, E., Tesei, L.: BioShape: a spatial shape-based scale-independent simulation environment for biological systems. In: *ICCS 2010: Proc. of Simulation of Multiphysics Multiscale Systems*, 7th International Workshop (2010), <http://cosy.cs.unicam.it/bioshape/iccs2010.pdf>
15. CxALite, <http://github.com/paradigmatic/CxALite/>
16. Bartocci, E., Corradini, F., Di Berardini, M.R., Merelli, E., Tesei, L.: Shape Calculus. A spatial calculus for 3D colliding shapes. Technical Report 6, Department of Mathematics and Computer Science, University of Camerino (January 2010) (accepted for publication in the *Scientific Annals of Computer Science*) (to appear in 2010)
17. Evans, D.J.W., Lawford, P.V., Gunn, J., Walker, D., Hose, D.R., Smallwood, R.H., Chopard, B., Krafczyk, M., Bernsdorf, J., Hoekstra, A.G.: The application of multiscale modelling to the process of development and prevention of stenosis in a stented coronary artery. *Phil. Trans. R. Soc. A* 366(1879), 3343–3360 (2008)
18. Caiazzo, A., Falcone, D.E.J., Hegewald, J., Lorenz, E., Stahl, B., Wang, D., Bernsdorf, J., Chopard, B., Gunn, J., Hose, R., Krafczyk, M., Lawford, P., Smallwood,

- R., Walker, D., Hoekstra, A.: Towards a Complex Automata Multiscale Model of In-Stent Restenosis. LNCS, vol. 5544, pp. 705–714. Springer, Heidelberg (2009)
19. Christley, S., Zhu, X., Newman, S.A., Alber, M.S.: Multiscale agent-based simulation for chondrogenic pattern formation in vitro. *Cybern. Syst.* 38(7), 707–727 (2007)
 20. Mullender, M.G., van der Meer, D.D., Huiskes, R., Lips, P.: Osteocyte density changes in aging and osteoporosis. *Bone* 18(2), 109–113 (1996)
 21. Marotti, G., Cané, P.S., Palumbo, C.: Structure-function relationships in the osteocyte. *Ital. J. Miner. Electrolyte Matab.* 4, 93–106 (1990)
 22. Penninger, C., Patel, N., Niebur, G., Tovar, A., Renaud, J.: A fully anisotropic hierarchical hybrid cellular automaton algorithm to simulate bone remodeling. *Mechanics Research Communications* 35(1-2), 32–42 (2008)
 23. Corradini, F., Merelli, E.: Hermes: agent-base middleware for mobile computing. In: Bernardo, M., Bogliolo, A. (eds.) SFM-Moby 2005. LNCS, vol. 3465, pp. 234–270. Springer, Heidelberg (2005)
 24. Textor, J., Hansen, B.: Hybrid simulation algorithms for an agent-based model of the immune response. *Cybern. Syst.* 40(5), 390–417 (2009)

CANv2: A Hybrid CA Model by Micro and Macro-dynamics Examples

Claudia R. Calidonna¹, Adele Naddeo²,
Giuseppe A. Trunfio³, and Salvatore Di Gregorio⁴

¹ CNR-ISAC Istituto di Scienze dell'Atmosfera e del Clima, Area Industriale,
Comparto 15 - 88046 Lamezia Terme (CZ), Italy

² CNISM, Unità di Ricerca di Salerno and Dipartimento di Fisica "E. R. Caianiello",
Università degli Studi di Salerno, 84084 Fisciano (SA), Italy

³ Facoltà di Architettura, Università degli Studi di Sassari, 07041 Alghero (SS)

⁴ Italy Dipartimento di Matematica, Università della Calabria,
87036 Rende (CS), Italy

cr.calidonna@isac.cnr.it, naddeo@sa.infn.it,
trunfio@uniss.it, toti.dig@unicall.it

Abstract. Cellular Automata (CA), one of the most challenging computational paradigms in microscopic and macroscopic complex systems simulation, can be successfully addressed also by using a modified CA classical approach. In this contribution we discuss related aspects in applying the CANv2 approach in examples of micro and macro dynamics such as: superconductive devices and forest fire simulation. Advantages and limitations are introduced when both microscopic and macroscopic dynamics are taken into account justifying the introduction of hybrid components between single cellular automata, i.e. a network in which global behavior and local interactions can coexist with side effects in computational parallelism addressing.

1 Introduction

Today Cellular Automata (CA) are a powerful and reliable approach, alternative to differential equations, for modeling and simulation of complex dynamical systems, whose evolution can be described by considering only local interactions between their elementary parts. So, the strategy is the decomposition of a complex phenomenon into a finite number of elementary processes, successfully applied by Di Gregorio [1] method, the overall dynamics being the combination of such elementary processes. Thus CA provide useful models for a lot of applications in natural sciences, ranging from the simulation of fluid dynamics to physical, chemical and geological processes.

Our hybrid computational model, the Cellular Automata Network version 2 (CANv2) model, has been introduced [2], which is particularly suitable for the simulation of microscopic as well as macroscopic phenomena, introducing the possibility to have a hybrid network of standard cellular automata components and global operators. Each automaton of the network represents, for instance, a

component of the physical system to be simulated and the connections among the network automata represent a disjoinable evolving law which characterizes the evolution of the physical system. That makes possible to exploit two different types of parallelism: a data parallelism, which comes from the use of CA classical model, and a task parallelism, which could arise when considering the network of CA. Furthermore the presence of global operators is mandatory when an external influence, to the macroscopic complex phenomena, assumes a control role and allows for expressing any mechanism that could not be expressed in terms of local interactions. These mechanisms could not influence the evolution of the phenomenon.

The aim of this contribution is to show, through concrete examples, how the CANv2 approach works as a tool to model and simulate the dynamics of a wide range of phenomena in natural science. In the following Section the CANv2 network is introduced and its power in simulating micro- and macroscopic dynamics is fully exploited by focusing on two examples: superconductive devices [2] and forest fires [3]. Finally some conclusions and perspectives of this work are outlined.

2 CANv2: A Hybrid Paradigm for Complex Systems Simulation

In this Section we introduce the CANv2 approach and show how it works as a simulation tool for complex systems, where the dynamical evolution is characterized by more than one mechanism and different scales. We focus on two concrete examples, borrowed from both the realm of natural macroscopic phenomena and that of condensed matter physics and microelectronics respectively, whose complex dynamic evolution depends on local interactions between the components: the operation of a topologically protected qubit based on a superconductive device and the spread of a forest fire.

A CANv2 network in a one dimensional cellular space is the following tuple:

$$\langle \mathcal{L}, X, S, G, P, P_{var}, f, g, F \rangle \quad (1)$$

- $\mathcal{L} = \{x : x \in \mathcal{N}, 0 \leq x \leq l_x\}$ is the set of points with integer coordinates in the finite region where the system evolves, \mathcal{N} is the set of natural numbers, and l_x is the upper bound of the set of points, i.e. it determines the bounds of the region of the system evolution.
- $X^{\#N_{tot}} = \bigcup_{i=1}^M X^{\#N_i}$, where M is the number of components, $\#N_{tot}$ is the cardinality of the total neighborhood set, and $X^{\#N_i} = [i - r, i + r] \subset \mathcal{L}$, is the set which identifies the geometrical pattern of cells which influences the cell state change, i.e. the neighborhood set for each cell i , where r is the radius.
- S is a finite set of states, where $S = X_{i=1}^{m_s} S_i$ is the Cartesian product of all the sets of sub-states and m_s is the total number of such sets.
- G is a finite set of global variables, where $G \subseteq \mathcal{R}$ (\mathcal{R} is the set of real numbers).

- P is a finite set of parameters, where $P \subseteq \mathcal{R}$.
- P_{var} is the set of global parameters.
- f is the set of the cellular automata transition functions.
- g is the set of global operator functions.
- $F : S^{\#N_{tot}} \times G \times P \times P_{var} \rightarrow S \times G \times P_{var}$ is the general transition for all the cells in \mathcal{L} . (\times is the Cartesian product)

Complex systems, which are made of different components, give rise to different transition functions, i.e. different cellular automata nodes for the network. In such a case all the components can be enumerated according to two main driving procedures: dependence relation constraints and consequent sequential transition function application respectively. Sometimes, when a direct dependence is not present a different sequence of application of the transition function between components can be adopted. In order to simplify the definition of the total transition function for the whole system, it could be useful to separately define the CA component and the global operator components as follows. Let $f_1 \cdots f_M$ be the transition functions for a system made of M components; the total transition function of the network, denoted with f , can be expressed in terms of a composition law such as:

$$f = f_1 \circ f_2 \circ \cdots \circ f_M \quad (2)$$

where \circ is the composition operator. Furthermore, from an execution point of view any transition function must be executed before the M -th.

Let $g_1 \cdots g_K$ be global operators for the system made of K components, then the global function g of the network can be expressed in terms of a composition law as follows:

$$g = g_1 \circ g_2 \circ \cdots \circ g_K \quad (3)$$

where \circ is the composition operator. From an execution point of view, if relation (3) holds then any global operator, but K -th, must be executed before the last one.

Let A be a cellular automaton, whose property is denoted by \mathbf{p} and transition function by \mathbf{f} , and \mathbf{GO} a global operator, whose global variable is denoted by g_v , and whose function is denoted by \mathbf{g} . If A needs to know the value/s of a global variable g_v at the same macro-step, defined as the whole network time step evolution, in order to evolve at each micro-step, defined as the time step evolution for each component cellular automaton or global operator node, then the execution of the \mathbf{g} function must precede the execution of the transition function \mathbf{f} . So, merging the previous definitions we obtain the following

$$F : g \circ f \quad (4)$$

$$F : \circ_{i,j=1}^{M,K} [g_j] \circ [f_i] \quad (5)$$

where the $[g_j]$ ($[f_i]$) symbol implies that its presence could be optional and the corresponding precedence relations must be expressed according to it.

The requirement of introducing components and their composition mechanism is needed in order to have the possibility to represent a complex system as a composition of more than one cellular automaton or global operators. In the CANv2 model this is made possible through the abstraction of a network of cellular automata. A network of cellular automata can be represented as a graph, the CAN precedence graph; here nodes represent cellular automata or global operators components while edges represent the precedence relations between nodes.

In the following let us show how the CANv2 paradigm works by making explicit reference to two applications in different contexts. In particular the example within the forest fire context is a re-elaboration of the original forest fire CA [3] model, developed by application of the Di Gregorio and Serra [1] methodology. We point out that a similar analysis has been carried out for the case of a debris flow after a landslide, by generalizing the original SCIDDICA [4] model in order to build up a network of cellular automata [5].

2.1 Superconductive Devices

In this Subsection we apply the CANv2 approach to build up a qualitative model of a topologically protected qubit [2], physically realized with a fully frustrated Josephson junction ladder (JLL) arranged in an annular geometry (see Ref. [6] for details). In its simplest version such a device consists of a ladder with N plaquettes closed in a ring with a half flux quantum ($\frac{1}{2}\Phi_0 = \frac{1}{2}\frac{hc}{2e}$) threading each plaquette. The number of plaquettes must correspond to the number of junctions on the vertical links of the ladder, so that each plaquette contains two junctions on the left and right link respectively. If we consider a ladder with, say, $N = 10$ plaquettes, the corresponding ground state is twofold degenerate with antiferromagnetic ordering as a result of superconductive currents circulating clockwise and counterclockwise in odd and even plaquettes respectively. That makes possible a mapping into a linear antiferromagnetic chain of half-integer spins. Such a two-state device can be manipulated by performing an adiabatic change of local magnetic fields. The tunneling between the two ground states $|0\rangle, |1\rangle$ corresponds to the process of creation and annihilation of kink-antikink pairs [6], which gives rise to a sequence of double flips in the spin pattern. A kink-antikink pair can be produced increasing the local magnetic field, that is applying a frustration single sawtooth pulse. Our device has N degrees of freedom and $\frac{N}{2}$ double flips are needed to pass from $|0\rangle$ to $|1\rangle$. But there are in general $(\frac{N}{2})!$ paths along which such processes can occur. Then, switching off the frustration the system relaxes on the new state and the transition is carried out. Based on such considerations, we construct an elementary NOT gate by setting up a quasi-probabilistic model which implements logical reversibility [2].

The starting point are the following identifications: 1) a grid, 2) a boundary condition, 3) each plaquette with each cell grid, 4) the phase differences values at each cell side for the device initialization, 5) a set of parameters, such as the current, the capacitance and so on, 6) a frustration single sawtooth magnetic pulse B . Given the considerations above, see Refs. [2] for details, the formal definition of the CANv2 model for the JLL device under study follows:

$$\langle \mathcal{L}, X, S, G, P, Pvar, f, g \rangle \quad (6)$$

- $\mathcal{L} \subseteq \mathcal{N}$ is the set of integer points in the finite region, the array, where the system evolves; each point identifies a cell. The lattice grid is a linear array with, for example, $\#\mathcal{L} = 10$ cells.
- X is the neighborhood set $[x - 2, x + 2]$ for each cell x .
- $S = S_1 \times S_2 \times S_3 \times S_4$ is the set of state values; it is specified by the cartesian product of the finite sets of values of the four following sub-states:
 1. Pseudo_S, pseudospin assuming values from $\{-1, 1\}$,
 2. Mp, magnetic pulse for each cell, fixed and invariant for each time step,
 3. LABEL, cell label in order to identify the cell, corresponding to a monotonic enumeration for all the cells, itself invariant for each time step.
 4. FLIP is the flip state in order to register if pseudospin flipping has been taken.
- G is the set of global variables: B_{tot} is the total applied magnetic pulse and $Start_c$ is the number of the corresponding starting cell.
- P is the finite set of global constant parameters: the current I inside the cell and the capacitance C .
- $Pvar$ is the finite set of the CA component variables: STEP is the step iterator which allows to trigger the evolution.
- $f : S \rightarrow S$ is the deterministic state transition for the determination of the pseudospin state and values.
- $g : S_2 \rightarrow G$ expresses the global operator which controls the total magnetic pulse applied on the system.
- $g_s : G \rightarrow G$ is the global operator which chooses randomly the driving cell.

In view of the implementation of a protected qubit the boundary condition topology is annular. In order to get a transition between the two ground states the magnetic pulse period must be related to the CA time step and it is equal to the pulse period, in order to capture the maximum pulse value. In order to implement the flipping procedure from $|0\rangle$ to $|1\rangle$ our model will select out one particular path (out of $(N/2)!$ paths), so giving rise to a high level description of such tunneling processes: in fact, at this preliminary stage we are interested only in the net result, i.e. the *NOT* operation. For this reason we choose to use a double step for the CA transition component, with each time step equal to the half of the single sawtooth magnetic pulse period. The CA component has, as initial condition, the pseudospin configuration obtained in the precedent stage since it must obey to an antiferromagnetic arrangement and the flipping state is zero. At the initial time, our device is in a steady state, in one of the two possible ground states. Each parameter is fixed and the LABEL values are fixed for all transition steps, but the variable STEP is initialized to each macro-step T. The studied model reproduces the *NOT* operator in more detail: together with the basic behavior, one of the possible flipping procedures is taken into account.

The general transition function takes into account the coupling factor, adding an external frustration, as a single sawtooth magnetic pulse acting on each lattice

```

Operator PULSE
BEGIN
  Btot:=0
  FOR i=1 TO N
    Btot:=Btot+Mp(i)
  ENDOFOR
  IF Btot<N*Mp(i) THEN
    error
  ENDIF
END

Operator CHOOSER
BEGIN
  StartC:=random(even(N))
END

Transition Function CA
BEGIN
  STEP:=STEP+1
  IF Mp(i)>0 THEN
    IF (LABEL(i)=StartC AND
        LABEL(i+1)=StartC+1) AND
        (FLIP(i+1)=0 AND
         FLIP(i+2)=0 AND
         FLIP(i-2)=0 AND
         FLIP(i-1)=0))
      THEN
        Pseudo_3(i+1):=Pseudo_3(i)
        Pseudo_3(i):=-1-Pseudo_3(i+1)
      ENDIF
    ELSE
      IF (FLIP(i)=0 AND
          FLIP(i-1)=1 AND
          FLIP(i-2)=1 AND
          LABEL(i) mod 2 = 0)
        THEN
          Pseudo_3(i+1):=Pseudo_3(i)
          Pseudo_3(i):=-1-Pseudo_3(i+1)
        ENDIF
        LABEL(i):=LABEL(i)+2
        CALL Transition Function CA
          until STEP not equal N/2
      ENDIF
    END
  END
END

```

Fig. 1. The JLL system components according to a CANv2 simulation

cell. The system transition is assumed to be given by the possible simultaneous applications of the two global operators followed by the transition function. The evolution of the model obeys to the following function $F : S_f \times G \times P \times Pvar \rightarrow S_f \times G \times Pvar$. The transition function scheme (see Fig. 1) shows two global operators, the pulse application and the random starting cell chooser, and the transition function repeatedly applies the cellular automata components according to the multiplicity related to the double flips. The probabilistic behavior is due to the Operator CHOOSER which randomly chooses at each network time step the driving cell for the flipping procedure. Such an operator is equipped with a global vector (a global variable which is able to store more values) which stores at each time step the starting cell for the evolution. In this way it is much simpler to recover the previous values as, for each time step directly accessible, the first evolution cell can be recovered.

2.2 Forest Fires

In this section we deal with a forest fire model. According to the forest fire model [3] the phenomenon is a two-dimensional CA with hexagonal cells and a radius two neighborhood. The lattice expressed by CAN method remains the same that in the original ones.

Also the set of parameters $P = \{p_e, p_s, p_v, p_t, p_w, p_{wd}, p_{wdr}\}$ such as (apothem of the cell, time step, type of vegetation and related catch burning, current time, weather condition depending on exposition season and atmospheric parameters and, finally, free wind direction and rate) is the same. The finite set S of the states of the cell is: $S = S_A \times S_V \times S_T \times S_H \times S_C \times S_D \times S_{WD} \times S_{WR} \times S_{FS}^{18} \times S_{FA}^{18}$ where the substates are

- S_A , *altitude*, takes the altitude value of the cell;
- S_V , *vegetation*, specifies the type of vegetation, relatively to the properties of catching fire and burning (see [3] for the specification of the value);

- S_T , *temperature*, takes the temperature value of the cell;
- S_H , *humidity*, takes the relative humidity value of the cell;
- S_C , *combustion* for the two possible fire types in the cell: *surface fire* and *crown fire*. It takes one of the values *not-inflammable*, *inflammable*, *burning* and *burnt* for each type;
- S_D , *duration*, takes the value of the duration of the fire in the cell;
- S_{WD} , *wind direction*, takes the values of the wind directions (the eight directions of the wind rose) at the ground level (that could be different from the free wind direction);
- S_{WR} , *wind rate*, takes the values of the wind rate (Km 0-0) at the ground level (that could be different from the free wind rate);
- S_{FS} , *fire spread*, accounts for the fire spread from the central cell to the other neighboring cells;
- S_{FA} , *fire acquire*, is just a renaming of S_{FS} and individuates the fire propagation from the other neighboring cells towards the central cell.

For the moment let us postpone the discussion regarding the mapping of substates.

The application of external influence regards some cells where the fire starts and some cells in which there is the intervention of the fireman. When considering cells, or a set of them, in which the fire starts, this interests a subregion of the entire lattice as $I \in K$. This is translated in the CAN model as a global operator and, in such a case, the application of the owner rule introduces much more specifications regarding the substate S_C . The function $\Gamma = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$, which is computed at each step, before the application of the transition function, represents the external influence.

- γ_1 determines the current time of the CA step;
- γ_2 supplies the weather conditions related to sun, wind direction and its rate, at each CA step;
- $\gamma_3 : I \rightarrow S_C$ accounts for external setting fire to cells of I at prefixed steps;
- $\gamma_4 : K \rightarrow S_C$ accounts for firemen intervention at prefixed steps.

The transition function is defined as: $\sigma = \sigma I_3 \otimes \sigma I_2 \otimes \sigma I_1 \otimes \sigma T_1$.

First the following internal transformation, concerning the effects of combustion in surface and crown fire inside the cell, is computed:

$\sigma T_1 : S_V \times S_T \times S_H \times S_C \times S_D \times S_{WD} \times S_{WR} \rightarrow S_T \times S_H \times S_C \times S_D$. Then the following local interactions are applied:

- $\sigma I_1 : (S_A)^{19} \rightarrow S_{WD} \times S_{WR}$
- $\sigma I_2 : (S_{FA})^{18} \times S_C \rightarrow S_C$
- $\sigma I_3 : (S_A)^{19} \times S_V \times S_C \times S_H \times S_T \times S_D \times S_{WD} \times S_{WR} \rightarrow (S_{FS})^{18}$

where σI_1 computes the wind direction and rate at the cell altitude, σI_2 computes the change of combustion conditions in the current cell and σI_3 computes the fire spread toward the neighboring cells.

- **Internal transformation.** σ_{T_1} acts when doesn't change substate S_D if the substate S_C is not *burning*, while the substates S_H and S_T vary their previous values on the basis of the weather change (p_w), the day hour (p_t), the wind (S_{WD} and S_{WR}) and the vegetation type (S_V and P_v). When the substate S_C is *burning*, then the substates S_T , S_H , S_C and S_D depend on the previous values of S_H , S_D and S_T . The value 0 for S_D determines the change of S_C to *burnt*. The conditions for the ignition from fire surface to crown fire are applied in T_1 .
- **Local interaction.** I_1 computes the substates S_{WD} and S_{WR} , *wind direction and rate* depend on pwd and pwr that represent the values of the free wind. The new values of S_{WD} and S_{WR} are obtained by adding the altitude vector to the corrective vector [3].
- **Local interaction.** I_2 tests if the fire is spreading toward the central cell starting from the other cells of the neighboring. If the combustion substate S_C is *inflammable*, then it changes to *burning*.
- **Local interaction.** I_3 computes the spread of fire acting if the state S_C is *burning*; the following computation steps, depending on the substates $S_V, S_H, S_T, S_D, S_{WD}, S_{WR}$ and the set of parameters P_v , are considered according to the maximum spread vector, determined as the sum of the slope effect and the wind effect, computing maximum spread (R_{max}), and considering an ellipse with a focus in the center of the cell whose area depends on R_{max} and such that the fire can propagate towards the neighboring cells inside the ellipse, i.e. S_{FS} (which is an alias of S_{FA}) takes the value *true*.

In these considerations all the substates of the forest fire CA become properties of the CAN network model but dealing with substate S_C requires much more attention because, according to the owner rule in CAN, it violates this principle as the substate, i.e. the property, is written by the external transformations γ_3 and γ_4 and internal transformation and local interaction σ_{I_2} . At this stage subprocesses must be considered. Regarding γ_1 that determines the current time of the CA step, in our case the global operator GO_1 determines the macro time step for all global operators and cellular automata components. On the basis of the same considerations, GO_2 supplies for the weather global condition for each macro step. When considering γ_3 , it is possible to introduce the property S_B *burning* as boolean values as it should be assumed as the state is burning or not and it result modified as $GO_3 : I \rightarrow S_B$. According to γ_4 , it is possible to introduce the property S_{Foff} *Fire off* as boolean values because the state is fire switched off or not and it results modified as $GO_4 : I \rightarrow S_{Foff}$.

As a first result all global operators could be computed simultaneously because they don't violate any access in the memory as in the original model for the substates i.e. the memory location of S_C .

And the automaton A_1 has the following transition function:

$$f_1 : S_V \times S_T \times S_H \times S_D \times S_{WD} \times S_{WR} \times S_C \times S_B \times S_{SB} \times S_{Foff} \rightarrow S_T \times S_H \times S_B \times S_D$$

where for us S_C , the property *combustion*, takes the possible values: *surface fire* and *crown fire*; it takes one of the values *not-inflammable*, *inflam-mable*.

The local interaction doesn't change as it maps exactly to the automaton A_2 with the transition function corresponding exactly to $\sigma_{I_1} : f_2 : (S_{FA})^{18} \times S_C \rightarrow S_C$.

Regarding the local interaction σ_{I_2} , a further substate, for us a property, S_{SB} *sensible to burn* should be introduced when we consider: if the cell is burning or not, or, if the fire was switched off, the weather conditions, combustible type and fire duration, all that establishes if the cell is *sensible to burn* according a threshold turn-on.

Then A_3 has the following transition function: $f_3 : (S_{FA})^{18} \times S_B \times S_C \rightarrow S_C$, A_4 has the following transition function: $f_4 : (S_A)^{19} \times S_V \times S_C \times S_H \times S_T \times S_D \times S_{WD} \times S_B \times S_{SB} \times S_{Foff} \times S_{WR} \rightarrow (S_{FS})^{18}$.

Finally the entire process could be summarized in three main stages following the corresponding order (where a component in the same stage can be computed at the same time) : **global influence** (GO_1, GO_2, GO_3, GO_4), **spreading condition** (A_2 and A_3), **burning condition change** A_1 , and **fire spreading toward their neighborhood** A_4 ; a graphical sketch is given in the figure below.

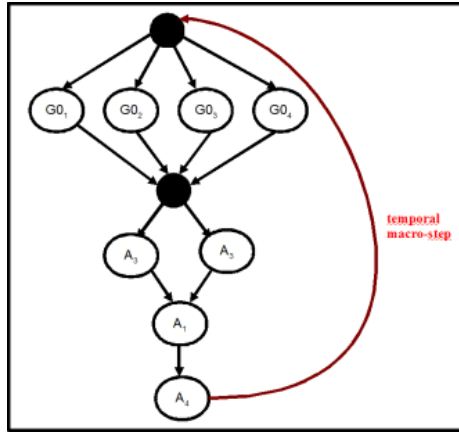


Fig. 2. CANv2 resulting graph for forest fire example mapped from the original forest fire CA model

3 Conclusions

In this paper we deal with CANv2 methodology in applying a hybrid CA computational model for micro and macro simulation. As a macro dynamics example we choose one application modeled according to the Di Gregorio [1] methodology. In this last case general characteristics for the mapping between the original and our model arise:

1. the region of interest remains the same;
2. the time step for the original CA becomes the macro step for the mapped CA network;

3. the mapping is the final result of a sort of heuristic refinement method with respect to the identification of local interactions and internal transformations where:
4. the definition and mapping of the neighbor set, the transition function, the substates set and the properties set require accurate considerations,
5. both internal transformation and local interaction generally individuate a CAN cellular automaton,
6. in order to satisfy the owner rule, in some cases sub-elementary processes of the original model should be introduced and consequently more transition functions and sometimes more properties; this results in more than one component in CAN and, as a consequence, more cellular automata and properties should be defined,
7. each external influence individuates a global operator.

Using such an approach has the disadvantage that, when modeling, the definition of processes and sub-processes to introduce in order to obey to the CAN owner rule for the properties components, results much more complex. Advantages arise because a first computational and load balancing scheme is already adopted at modeling stage, as at run-time the resulting application can exploit much more parallelism resources than in a classical CA scheme model. Next research step will focused on implementation aspect and demonstrate real advantages in parallel execution.

Acknowledgements

This research was partially funded by ASI (Italian Space Agency) project - “SARFIRE: Spaceborn SAR imagery and environmental data fusion for the dynamical evaluation of land regions susceptibility to fire” ID: 2288. Main AB-BAMPAU applications for forest fires will be based on ASI satellitar images of COSMO-SkyMed program.

References

1. Di Gregorio, S., Serra, R.: An empirical method for modelling and simulating some complex macroscopic phenomena by cellular automata. *Fut. Gen. Comp. Syst.* 16, 259–271 (1999)
2. Calidonna, C.R., Naddeo, A.: Addressing reversibility in quantum devices by a hybrid CA approach: the JJI case. *Int. J. Unconv. Comp.* 4, 315–340 (2008)
3. Trunfio, G.A.: A Predicting Wildfire Spreading Through a Hexagonal Cellular Automata model. In: Sloot, P.M.A., Chopard, B., Hoekstra, A.G. (eds.) *ACRI 2004. LNCS*, vol. 3305, pp. 385–394. Springer, Heidelberg (2004)
4. D’Ambrosio, D., Di Gregorio, S., Iovine, G., Lupiano, V., Longo, R., Spataro, W.: First simulations of the Sarno debris flows through Cellular Automata modelling. *Geomorphology* 54, 91–117 (2003)
5. Calidonna, C.R., Di Gregorio, S., Mango Furnari, M.: Mapping applications of cellular automata into applications of cellular automata networks. *Comp. Phys. Comm.* 147, 724–728 (2002)
6. Cristofano, G., Marotta, V., Naddeo, A., Niccoli, G.: A conformal field theory description of magnetic flux fractionalization in Josephson junction ladders. *Eur. Phys. J. B* 49, 83–91 (2006)

Simulation of Traffic Flow at a Signalised Intersection

Somayyeh Belbasi and M. Ebrahim Foulaadvand

Department of Physics, Zanjan University, P.O. Box 45196-313, Zanjan, Iran

Abstract. We have developed a Nagel-Schreckenberg cellular automata model for describing of vehicular traffic flow at a single intersection. A set of traffic lights operating either in fixed-time or traffic adaptive scheme controls the traffic flow. Closed boundary condition is applied to the streets each of which conduct a uni-directional flow. Extensive Monte Carlo simulations are carried out to find the model characteristics. In particular, we investigate the dependence of the flows on the signalisation parameters.

1 Introduction

Modelling the dynamics of vehicular traffic flow by cellular automata has constituted the subject of intensive research by statistical physics during the past years [1,2,3]. *City traffic* was an early simulation target for statistical physicists [4,5]. Evidently the optimisation of traffic flow at a single intersection is a preliminary but crucial step to achieve the ultimate task of global optimisation in city networks [6]. In principle, the vehicular flow at the intersection of two roads can be controlled via two distinctive schemes. In the first scheme, the traffic is controlled without traffic lights [7]. In the second scheme, signalised traffic lights control the flow. Our objective in this paper is to study in some depth, the characteristics of traffic flow and its optimisation in a single intersection with closed boundary condition.

2 Description of the Problem

Imagine two perpendicular one dimensional closed chains each having L sites and unidirectional vehicular traffic flows. They intersect each other at the middle sites $i_1 = i_2 = \frac{L}{2}$ on the first and the second chain. With no loss of generality we take the flow direction in the first chain from south to north and in the second chain from east to west. (see Fig.1 for illustration). Cars are not allowed to turn. Each car occupies an integer number of cells denoted by L_{car} . Time elapses in discrete steps of Δt and velocities take discrete values $0, 1, 2, \dots, v_{max}$ in which v_{max} is the maximum velocity. To be more specific, at each step of time, the system evolves under the Nagel-Schreckenberg (NS) dynamics [8]. The length of each car is taken 4.5 metres. Therefore, the spatial grid Δx (cell length) equals to $\frac{4.5}{L_{car}} m$. We take the time step $\Delta t = 1 s$. Furthermore, we adopt a speed-limit

of 75 km/h . In addition, each discrete increments of velocity signifies a value of $\Delta v = \frac{4.5}{L_{car}} \text{ m/s}$ which is also equivalent to the acceleration. Moreover, we take the value of random breaking parameter at $p = 0.1$.

3 Fixed Time Signalisation of Lights

In this scheme the period T , *cycle time*, is divided into two phases. In the first phase with duration T_g , the lights are green for the northward street and red for the westward one. In the second phase which lasts for $T - T_g$ timesteps the lights change their colour. The gap of all cars are update with their leader vehicle except those two which are the nearest approaching cars to the intersection. For these approaching cars gap should be adjusted with the signal in its red phase. The streets sizes are $L_1 = L_2 = 1350 \text{ m}$ and we take $L_{car} = 5$. The system is update and after transients, two streets maintain steady-state currents denoted by J_1 and J_2 which are defined as the number of vehicles passing from a fixed location per time step. In general, the dependence of total current on ρ_1 depends on the value of T_g . Except for small values of T_g , total current increases with ρ_1 then it becomes saturated at a lengthy plateau before it starts its linear decrease. We have also examined the behaviour of J_{tot} for other values of ρ_2 . Figures (2) exhibits the result for $\rho_2 = 0.05$. Our simulations confirm that for small ρ_2 up to 0.1 total current shows a distinguishable dependence on T_g in the entire range of ρ_1 especially in intermediate values. In contrast, for $\rho_2 > 0.1$, we observe no significant dependence on T_g in the intermediate ρ_1 but we observe notable dependence for large ρ_1 .

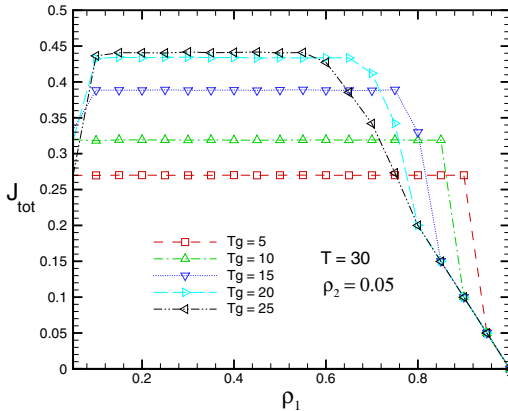


Fig. 1. Total current J_{tot} vs ρ_1 for various values of T_g at $T = 30$

4 Traffic Responsive Signalisation

In this section we present our simulations results for the *so-called* intelligent controlling scheme in which the traffic light cycle is no longer fixed [9,10], the

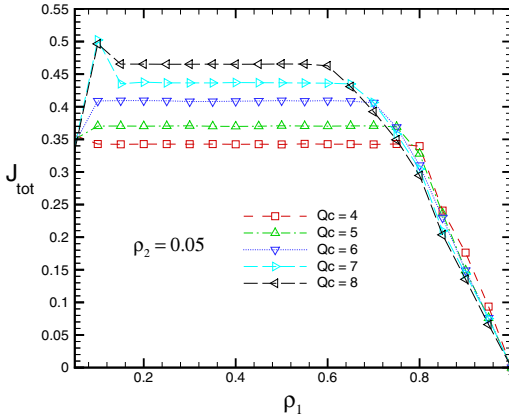


Fig. 2. Total current vs ρ_1 at $\rho_2 = 0.05$ for various values of cut-off lengths Q_c and ρ_2

signalisation of traffic lights is simultaneously adapted to traffic status in the vicinity of intersection. This scheme has been implemented in simulation of traffic flow at intersections with open-boundary conditions [11]. To be precise, we define a cut-off queue length Q_c . The signal remain red for a street until the length of the corresponding queue formed behind the red light exceeds the cut-off length Q_c . At this moment the lights change colour. Apparently due to stochastic nature of cars movement, the cycle time will be subjected to variations and will no longer remain constant. In figure (3) we exhibit J_{tot} versus ρ_1 . Analogous to fixed-time scheme, for given ρ_2 a lengthy plateau in J_{tot} forms. The plateau height as well as its length show a significant dependence on Q_c . higher Q_c are associated with smaller length and higher current. We have also examined larger values of ρ_2 . The results are qualitatively analogous the above graphs. The notable point is that for ρ_2 larger than 0.1, J_{tot} do not show a significant dependence on ρ_2 .

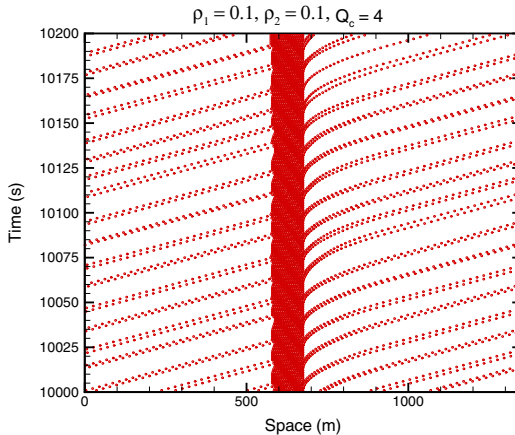


Fig. 3. Space-time plot of vehicles for traffic responsive schemes

To shed some light onto the problem, we sketch space-time plots of vehicles. It is seen that in traffic responsive scheme, the cars spatial distribution is more homogeneous which is due to randomness in cycle times. Lastly, we compare our results to those obtained in simulation of a nonsignalised intersection [7]. It can be concluded that signalisation strategies are apparently more efficient in comparison to non-signalisation scheme.

5 Summary and Concluding Remarks

By extensive Monte Carlo simulations, we have investigated the flow characteristics in a signalised intersection via developing a Nagel-Schreckenberg cellular automata model. We have considered two types of schemes: fixed-time and traffic responsive. In particular, we have obtained the fundamental diagrams in both streets and the dependence of total current on street densities. Our findings show hindrance of cars upon reaching the red light gives rise to formation of plateau regions in the fundamental diagrams. This is reminiscent of the conventional role of a single impurity in the one dimensional out of equilibrium systems. The existence of wide plateau region in the total system current shows the robustness of the controlling scheme to the density fluctuations.

References

1. Chowdhury, D., Santen, L., Schadschneider, A.: *Physics Reports* 329, 199 (2000)
2. Helbing, D.: *Rev. Mod. Phys.* 73, 1067 (2001)
3. Kerner, B.: *Physics of traffic flow*. Springer, Heidelberg (2004)
4. Biham, O., Middleton, A., Levine, D.: *Phys. Rev. A* 46, R6124 (1992)
5. Nagatani, T.: *J. Phys. Soc. Japan* 63, 1228 (1994); *J. Phys. Soc. Japan* 64, 1421 (1995)
6. Chituri, Y., Piccoli, B.: *Discrete and Continuous Dynamical Systems B* 5, 599 (2005)
7. Foolaadvand, M.E., Belbasi, S.: *J. Phys. A: Math, Theor.* 40, 8289 (2007)
8. Nagel, K., Schreckenberg, M.: *J. Phys. I France* 2, 2221 (1992)
9. Webster, F., Cobb, B.: *Traffic Signal Setting*. H.M.S Office, London (1966)
10. Bell, M.G.: *Transp. Res. A* 26(4), 303 (1992)
11. Foolaadvand, M.E., Sadjadi, Z., Shaebani, M.R.: *J. Phys. A: Math. Gen.* 37, 561 (2004)

A Novel Method for Simulating Cancer Growth

Mehrdad Ghaemi, Omid Naderi, and Zahra Zabihinpour

Chemistry Department, Tarbiat Moallem University, Tehran, Iran

mehrdad_ghaemi@hotmail.com

ph_ch_omid@yahoo.com

z_zabihin@tmu.ac.ir

Abstract. We propose probabilistic cellular automata on a square lattice for simulating the dynamic of cancer growth in a reaction-diffusion frame. In the reaction step each cancerous cell can proliferate, be quiescent, or die due to apoptosis or necrosis phenomenon. The three-state Potts model is used for calculating the probabilities in the reaction step. We consider the effect of nutrient in the tumor growth in order to improve the precision of the model. We use a simple and suitable method for the diffusion step to simplify movement of cells and nutrient in the model. In the diffusion step the lattice is partitioned by 3×3 blocks. In each block we count the number of different types of cells and redistribute them in the block. In the next time step, each block will be shifted one row down and one column to the right and the operation will be continued. The redistribution step for nutrient molecules is same as cells. It is shown tumor growths asymmetrically toward nutrient source. It has been shown such a simple model could simulate tumor growth with good accuracy, which is based on the well known physical ground i.e. the three-state Potts model.

1 Introduction

Cancer is a disease in which cells in body divide without any control. So the cells are not normal and could invade to other tissues of the body. They also may immigrate to other parts of the body through blood vessels and lymph. Although there are different kinds of cancers, most of cancers have some common characteristics. They are due to some mutations which lead to malignant tumors. Malignant tumors could invade and spread to other tissues. This process is called metastasis. When a tumor is going to grow, its size and number of cells will increase. So it will need nutrient to obtain energy. If it could not obtain sufficient nutrient, the cells will finally die. This process (dying cancer cells) is called necrosis [1]. Since the treatment of cancers is a challenging issue, different attempts is going to be considered. Study of tumor growth seems to be useful in understanding cancer in morphological and functional properties. The control of the interacting elements in a tumor is a difficult task in an experimental work. It is also difficult to predict the situation of the tumor growth since it is a biological complex system. Hence, Mathematical modeling using different methods could be helpful in understanding important features of such a complex system. Attempts such as using Ordinary or Partial Differential Equations (ODEs or PDEs) have been made for studying cancer and tumor growth [2-8]. Simulation methods like

Monte Carlo (MC) and Cellular Automata (CA) are also useful in which deterministic or probabilistic approaches are employed to obtain a final pattern. CA is a discrete mathematical method which is widely used from simple problems such as entertainment games to complex biological networks [9]. There are several kinds of CA with different abilities. One of them is the Lattice Gas Cellular Automata (LGCA) which mainly uses for the fluid diffusion processes [10]. Since simulation of cancer in most cases needs solving complex differential equations or including time consuming numerical methods, in this work we have tried to introduce a simple and efficient method to combine precision and physical aspect of the tumor growth. In a previous work, Ghaemi and Sahrokhi used combination of the LGCA and Cellular Potts Model (CPM) [11] for simulating tumor growth. The advantage of CPM is its ability for introducing cell-cell interaction in correct and well known physical way. Besides, LGCA can simulate movement of cell in simple and correct physical way. The disadvantages of their work were missing nutrition effect and a complicated unrealistic geometry (square lattice with five cells in each site and no interaction between adjacent sites). However, there are some features which should be considered to improve the simplicity and precision of the model. We have considered the effect of nutrient in the tumor growth in order to improve the precision of the model. In addition, we have used a simple and suitable method for the diffusion step to simplify the model. In the next section, we introduce the model and explain its details. Then, we show the results and discuss the advantage of the model over our previous method.

2 Method

The biological basis of the model consists of proliferation, motility, death, and competition between healthy and cancerous cells. Nutrients (such as oxygen, glucose, metal ions ...) diffuse to target through blood vessels. Therefore, there is a competition between cancerous and healthy cells in the case of nutrient limitation. A tumor consists of different types of cells, but in our model we only consider three types: healthy, cancerous, and necrotic cells.

The main body of the model is like reactive-diffusion systems. In the reaction step each cancerous cell could proliferate, be quiescent, or die due to apoptosis or necrosis phenomenon. Because there is lack of information about details of cell-cell interactions which arise from the complexity of the cell, it seems better to use probabilistic approach. We have used the CPM with Glauber algorithm for obtaining probabilities [12]. Coupling coefficients in the CPM could simulate the effect of cellular adhesive molecules present in the cell membrane. Assume $H_{i,j}$, $N_{i,j}$, and $C_{i,j}$ are the number of healthy, necrotic, and cancerous cells in four nearest neighbors of site (i,j) , respectively. So the configuration energy can be written as;

$$\frac{E_{\text{conf}}}{kT} = \sum_{i,j} E_{i,j} \quad (1)$$

where $E_{i,j}$ is the configuration energy of the site (i,j) and the sum is over the nearest neighbor sites. If site (i,j) is occupied by a cancer cell,

$$E_{i,j} = E_{i,j,\text{Cancer}} = -C_{i,j} K_{CC} \quad (2)$$

and if site (i,j) is occupied by a necrotic cell,

$$E_{i,j} = E_{i,j,\text{Necrose}} = -N_{i,j} K_{NN} \quad (3)$$

and finally, if site (i,j) is occupied by a healthy cell,

$$E_{i,j} = E_{i,j,\text{Healthy}} = -H_{i,j} K_{HH} \quad (4)$$

where K_{CC} , K_{NN} , and K_{HH} are coupling coefficients for cancerous-cancerous, necrotic-necrotic, and the healthy-healthy cells, respectively. Cell-cell interactions are adhesive, thus the couplings are positive (notice the minus sign in the equations 2-4). Equations 2-4 come from the configuration energy of the three-state Potts model [13]. Now in each lattice site one of the following reactions may take place at each time step:

1- If site (i,j) is occupied by a cancerous cell and if the concentration of nutrient is greater than 50% of initial concentration then: a) the cancerous cell can remain cancerous with probability $P_{\text{quiescent}}$, or b) will die according to necrosis with probability P_{necrosis} and it will be replaced with necrotic cell, or c) dies according to apoptosis with probability $P_{\text{apoptosis}}$ and it will be replaced with healthy cell. These probabilities are computed as follow;

$$P_{\text{apoptosis}} = \frac{\exp\{-E_{i,j,\text{Healthy}}\}}{\exp\{-E_{i,j,\text{Healthy}}\} + \exp\{-E_{i,j,\text{Necrose}}\} + \exp\{-E_{i,j,\text{Cancer}}\}} \quad (5)$$

$$P_{\text{necrosis}} = \frac{\exp\{-E_{i,j,\text{Necrose}}\}}{\exp\{-E_{i,j,\text{Healthy}}\} + \exp\{-E_{i,j,\text{Necrose}}\} + \exp\{-E_{i,j,\text{Cancer}}\}} \quad (6)$$

$$P_{\text{quiescent}} = \frac{\exp\{-E_{i,j,\text{Cancer}}\}}{\exp\{-E_{i,j,\text{Healthy}}\} + \exp\{-E_{i,j,\text{Necrose}}\} + \exp\{-E_{i,j,\text{Cancer}}\}} \quad (7)$$

However, if the concentration of nutrient is smaller than 50% of initial concentration the cancerous cell will die and it will be replaced by a healthy or necrotic cell.

2- If site (i,j) is occupied by a healthy cell, the concentration of nutrient is greater than 50% of initial concentration, and at least one of its nearest neighbors is a cancerous cell then proliferation can happen with probability $P_{\text{proliferation}}$ and the healthy cell will be replaced by a cancerous cell where;

$$P_{\text{proliferation}} = \frac{\exp\{-E_{i,j,\text{Cancer}}\}}{\exp\{-E_{i,j,\text{Healthy}}\} + \exp\{-E_{i,j,\text{Cancer}}\}}, \quad (8)$$

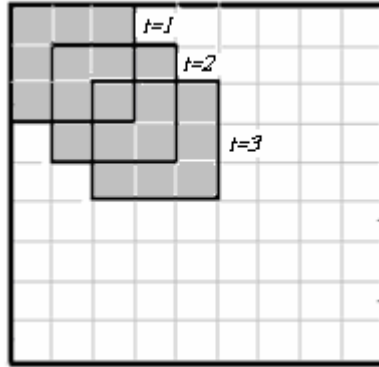


Fig. 1. The lattice is partitioned by 3×3 blocks. In each time step, the block will be shifted one row down and one column to the right.

otherwise the healthy cell remains healthy. Necrotic cells remain necrotic during simulation. In the diffusion (propagation and redistribution) step, each cell could move toward its neighbors. This step will be done based on a model which was first introduced by Vanag [14]. The lattice is partitioned by 3×3 blocks (Fig. 1). In each block we count the number of different types of cells. After that, cells will be redistributed in the block. The probability that one site will be occupied is proportional to the number of necrotic cells in its nearest neighbors in adjacent blocks. First, necrotic cells are redistributed because of their less ability to movement. Then, cancerous and healthy cells will be redistributed over remaining sites, respectively. By this rule, necrotic cells move to the core of tumor and remain near each others. In the next time step, the block will be shifted one row down and one column to the right and the operation will be continued (Fig. 1). The redistribution step for nutrient molecules is same as cells but all 9 sites in each block have equal probability to be occupied and there is no limitation on the number of nutrient molecules in each site. Since the rate of diffusion of nutrient is larger than cells we can do diffusion step for nutrient n times in each time step.

3 Result and Discussion

The simulation is conducted on a 400×400 square lattice with initially four cancerous cells in the center of lattice. According to the recent work of Gerlee et al [3], we set maximum length size of a tumor equal to $L=1$ cm so cancer cell area is about $6.25 \times 10^{-6} \text{ cm}^2$ which is in agreement with experiment. Initial glucose concentration is $1.3 \times 10^{-8} \text{ mol cm}^{-2}$ and the aerobic glucose consumption rate is considered $3.8 \times 10^{-17} \text{ mol cells}^{-1} \text{ s}^{-1}$. For simplicity, we used dimensionless number according to dimensionless partial differential equation in ref. [3] for diffusion and consumption of glucose. Each time step is rescaled to 16 hours which corresponds to the true proliferation age [3, 15, 16]. According to the eq.6 in ref. [3], in each lattice site non-dimensional glucose concentration varies from 0 to 1, and glucose consumption rate is 0.0075. For simplicity of algorithm we let glucose concentration varies from 0 to 400 so, glucose consumption rate is 3. Nutrient source is placed at the bottom of the lattice

in which nutrients diffuse through the tissue. We used a gradient of nutrient for the initial concentration in such a way that the nutrient concentration is 400 at the bottom of the lattice and 0 at the top of lattice. At each time step the concentration of glucose at bottom row is updated to 400. The rate of consumption of glucose is set to be 6 per time step for each cancerous cell (twice the normal cells). Threshold for glucose which induces necrosis or apoptosis is set to 50% of initial glucose concentration [17]. According to the work of Christley et al [18] the diffusion rate of cell is 1 pixels per 60 iteration and diffusion rate of chemicals is 28 -108 pixels per one iteration. In our model each pixels correspond to one cell and for each cell we have 400 boxes nutrient. We let cells diffuse in each time step so, according to above rescaled data we run diffusion step for nutrient approximately 7 times in each time step. Each site of the lattice just contains one of the three cell types. Necrotic cells are considered as death cells which do not consume nutrients. Multi cellular spheroids have a well-established characteristic structure. There is an outer rim of proliferating cells (a few hundred μm thick) and an inner core of necrotic cells. In between there is a layer of quiescent cells, which are not dividing but are alive, and can begin dividing again if environmental conditions change. The coupling parameters values taken as $K_{CC} = 3$, $K_{HH} = 0.5$ and $K_{NN} = 3$ are determined in such a way to produce multi cellular spheroids shape (Fig. 2). Figure 2 shows tumor growths asymmetrically toward nutrient source at the bottom of the lattice. Increasing the value of K_{HH} above 1.5 will lead tumor to be disappeared at a few time steps, because according to eq. 5 the probability of the apoptosis will increase rapidly. By increasing the value of K_{NN} the diameter of the layer of quiescent cells decreases more rapidly and simultaneously the rate of growing the inner core of necrotic cells increase. The future of tumor strongly depends on the values of K_{CC} and K_{NN} . For the values of $K_{CC}=K_{NN} < 1.5$ the tumor initially grows up and after some time steps the layers of proliferating and quiescent cell will be destroyed. The average number of cancerous cell versus time step is calculated for 20 different samples with the coupling parameters $K_{CC} = 3$, $K_{HH} = 1.5$ and $K_{NN} = 3$ (Fig. 3).

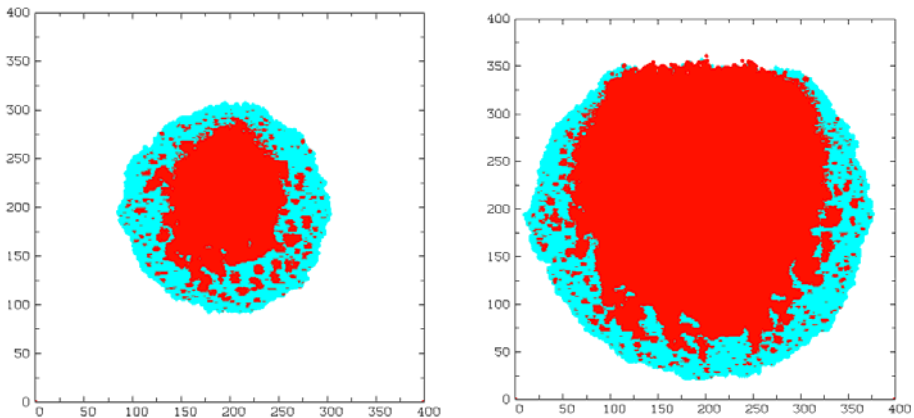


Fig. 2. Spatial distribution of the cells at $t = 60$ and 100 on a lattice of size 400×400 for $K_{CC} = 3$, $K_{HH} = 1.5$ and $K_{NN} = 3$. Necrotic cells in the core of tumor are shown in red, cancerous cells are light blue, and healthy cells are white.

After an initial exponential growth phase, number of cancerous cells increase with time with constant slope. The advantage of this method compared to other simulation of cancer growth is that the present method only needs three parameters K_{CC} , K_{HH} and K_{NN} which is based on the well known physical ground i.e. the three-state Potts model. The main aim of this work was to show the possibility of introducing movement of cells and nutrition effect in the cellular Potts model in a very simple and efficient way. The simulation has been greatly simplified by neglecting some crucial effects such as; the effect of oxygen and hydrogen concentrations and limited volume space for tumor. We expect addition of these effects may be introduced in the automata which is still under investigation.

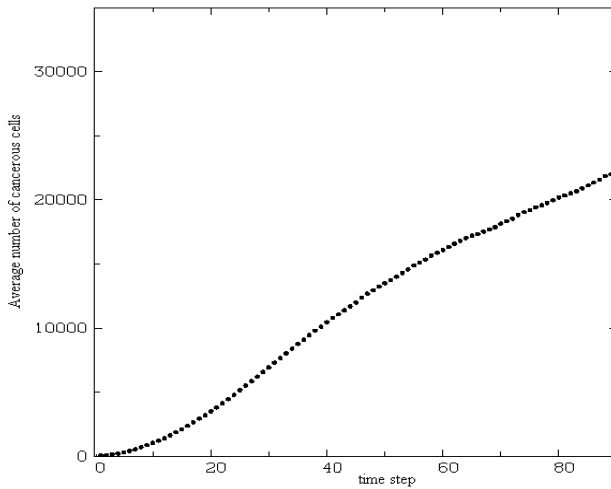


Fig. 3. The time evolution of average number of cancerous cells for $K_{CC} = 3$, $K_{HH}=1.5$ and $K_{NN} = 3$ on a lattice of size 400×400

References

1. Knowles, M.A., Selby, P.J.: Introduction to the Cellular and Molecular Biology of Cancer, 4th edn. Oxford University Press, Oxford (2005)
2. Matzavinos, A., Chaplain, M.A.J.: Mathematical modelling of the spatio-temporal response of cytotoxic T-lymphocytes to a solid tumour. *Mathematical Medicine and Biology* 21, 1–34 (2004)
3. Gerlee, P., Anderson, A.R.A.: An evolutionary hybrid cellular automaton model of solid tumor growth. *J. Theoretical Biology* 246(4), 583–603 (2007)
4. Khain, E., Sander, L.M.: Dynamics and pattern formation in invasive tumor growth. *Physical Review Letters* 96, 188103 (2006)
5. Marciniak-Czochra, A., Kimmel, M.: Reaction–diffusion approach to modeling of the spread of early tumors along linear or tubular structures. *J. Theoretical. Biology* 244, 375–387 (2007)
6. Ferreira, S.C., Martins, M.L., Vilela, M.J.: Reaction-diffusion model for the growth of avascular tumor. *Phys. Rev. E* 65, 021907 (2002)

7. Adam, J.A., Bellomo, N.: A Survey of Models for Tumor-Immune System Dynamics, Birkhäuser, Boston (1997)
8. Dormann, S., Deutsch, A.: Modeling of self-organized avascular tumor growth with a hybrid cellular automaton. *Silico Biology* 2, 0035 (2002)
9. Wolfram, S.: A new kind of science. Wolfram Media Inc., Champaign (2001)
10. Wolf-Gladrow, D.A.: Lattice-Gas Cellular Automata and Lattice Boltzmann Models: An Introduction. Springer, Heidelberg (2000)
11. Ghaemi, M., Shahrokhi, A.: Combination of the Cellular Potts Model and Lattice Gas Cellular Automata for Simulating the Avascular Cancer Growth. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 297–303. Springer, Heidelberg (2006)
12. Glauber, R.J.: Time-Dependent Statistics of the Ising Model. *J. Math. Phys.* 4, 294 (1963)
13. Wu, F.: The Potts-model. *Rev. Mod. Phys.* 54, 235–268 (1982)
14. Vanag, V.K.: Study of spatially extended dynamical systems using probabilistic cellular automata. *J. Physics-Uspekhi* 42(5), 413–434 (1999)
15. Piotrowska, M.J., Angus, S.D.: A quantitative cellular automaton model of in vitro multicellular spheroid tumour growth. *J. Theoretical Biology* 258(2), 165–178 (2009)
16. Calabresi, P., Schein, P.E.: Medical oncology, 2nd edn. Mc Graw-Hill, New York (1993)
17. Ganong, W.F.: Review of medical physiology, 19th edn. Appleton and Lang, New York (1993)
18. Christley, S., Zhu, X., Newman, S.A., Alber, M.S.: Multiscale Agent-Based Simulation for Chondrogenic Pattern Formation *In Vitro*. *Cybernetics and Systems: An International Journal* 38, 707–727 (2007)

Towards Cellular Automata Football Models with Mentality Accounting

Alexander Makarenko¹, Dmitry Krushinski², Anton Musienko¹,
and Boris Goldengorin²

¹Institute for Applied System Analysis at NTUU (KPI),
37 Pobedy Avenue, Kiev, 03056, Ukraine
makalex@i.com.ua

²University of Groningen, Faculty of Economics, P.O. Box 800 9700AV,
Groningen, The Netherland
b.goldengorin@rug.nl

Abstract. In this paper we deal with mathematical modeling of team sport games based on cellular automata (CA). We describe some developments of CA models of football. Presumable learning and optimization problems in team modeling based on CA are discussed. Some general problems are discussed which are related to the accounting of mentality of game participants.

Keywords: football, cellular automata, models, anticipation, mentality.

I Introduction

Recently the simulation of sport games (especially football) serves as a source for developing new approaches in understanding and modelling of games. Some aspects of collective sport games had been considered earlier: sport statistics; reinforcement learning; hardware implementations of players through robot teams and Championships on robot football (RoboCup) [1]; design of moving robots for competitions and many others. But all authors had stressed the great complexity and multi-aspect nature of the existing problems. Analysis of the demands on real world applications of CA follows to the new field of cellular automata using – namely modelling the evolution of collaborative teams of agents. As the main example of such system we consider the football. We propose some description of rules for modelling, development of models for evolution investigations, some examples of modelling and some ways for approach development.

2 Simple Football Model Based on Cellular Automata

Here we pose (only for outline of ideas) a very schematic description of CA model of football. The full description will be described in forthcoming publications.

First important assumption is that the game space is represented as the collection of cells just as in the cellular automata models for pedestrian movements. We consider

two teams of players. At each step we define all single available player movements within some fixed neighbourhood. In case of the Neumann's neighbourhood we compute the probabilities of player's movement into one of four neighbouring cells. Remark that we may accept extended neighbourhood for each of the players if we should take into account the different presumable velocities of players.

Each player is represented by occupied cell. The rules for player concerned the movement of player, operations with a ball and interactions between players. Only one player may be at any cell. The player also can move at the lattice step by step in vertical and horizontal directions. The rules for player's behavior in fact formalize the rules for decision of player in dependence on the situation at the field of game and in dependence of current score of the game. Also we set the probability to move for player to be zero if a neighbouring cell is occupied by a boundary, and assign non-zero probabilities to cells of all other directions. By increasing the probability in a chosen direction we model an intention of each player and the team to move simultaneously.

Proposed models had been realized as the special computer program for the modeling of game. For illustration we pose one of a computation examples.

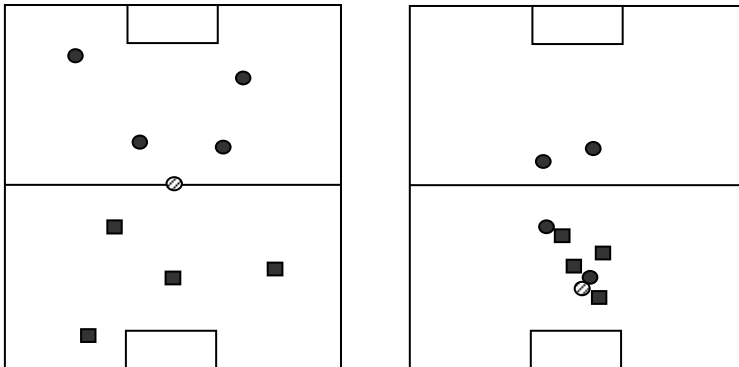


Fig. 1. At the left side of picture the example of beginning of the game situation is presented. The right side illustrates the approach of the player with the ball to the gate of opposite team after some time steps at game. (Each team has four players (black squares and black circles); the grey circle corresponds to the ball).

3 The Problems of Mentality Accounting in Game Simulation

In Sections 2 we have presented the classical approach of CA without taking into account the mentality properties for player movements. The accounting of mentality of the participants of social processes (including sport games) is one of the main tendencies in developing of more adequate models. There are many presumable ways of implementing such accounting – from the attempts to model the human consciousness and decision – making in artificial intelligence to the simplest statistical rules.

Of course many aspects related to the mentality accounting should be represented in the complicated models of the game: monitoring and recognition of game situation; decision – making process on movement direction, velocity and goals; possibilities of

movement implementation etc. [1]. Because of simple spatial structure CA models allow to implement a lot of properties of a team players in rather simple way.

Here we discuss the way for accounting one of the very important properties in space game namely anticipation in game. The anticipation property is that the individual makes a decision accounting the future states of the system [2].

One of the consequences is that the accounting for an anticipatory property leads to advanced mathematical models. Since 1992 starting from cellular automata the incursive relation had been introduced by D. Dubois for the case when 'the values of state $X(t+1)$ at time $t+1$ depends on values $X(t-i)$ at time $t-i$, $i=1,2,\dots$, the value $X(t)$ at time t and the value $X(t+j)$ at time $t+j$, $j=1,2,\dots$ as the function of command vector p ' [2]. In the simplest cases of discrete systems this leads to the formal dynamic equations (for the case of discrete time $t=0, 1, \dots, n, \dots$ and finite number of elements M):

$$s_i(t+1) = G_i(\{s_i(t)\}, \dots, \{s_i(t+g(i))\}, R), \quad (1)$$

where R is the set of external parameters (environment, control), $\{s_i(t)\}$ the state of the system at a moment of time t ($i=1, 2, \dots, M$), $g(i)$ horizon of forecasting, $\{G\}$ set of nonlinear functions for evolution of the elements states. "In the same way, the hyperincursion is an extension of recursion in which several different solutions can be generated at each time step" [2, p.98].

According [2] the anticipation may be of 'weak' type (with predictive model for future states of system, the case which had been considered by R. Rosen) and of 'strong' type when the system cannot make predictions.

Concerning the specific case of the game problems it had been recognised earlier that some kinds of anticipatory property is intrinsic for game. But in fact considered before in sport games anticipatory properties was of 'weak' type (with predictive models). The experience of investigations of models with anticipation – in game "Life" with anticipation [3] and especially in crowd movements [4] allows proposing some ways for further accounting of anticipation in team investigations.

At the local level each participant of the game process tries to anticipate the future state of game in local neighbourhood when he makes the decision on movement. Also the macro neighbourhood of game participants might be accounted for the common coach information. The adequate accounting of anticipatory property in the CA methodologies is a difficult problem because it requires also complication of CA models by introducing the internal states of CA cells and special internal dynamical laws for mental parameters. But just now we are able to propose some presumable consequences for game considerations. According [3, 4] the first step in anticipation accounting consist in the modification of CA rules by introducing formally the values in cells at next moment of time into the formal rules.

Of course the full implementation of such models is the task for further investigations but the extension of the results described in the present paper to new game models would open new possibilities for exploring behaviour of game participants. At first it may be considered the problem of accounting the different player's visions of the field of game by different neighbourhoods for each player.

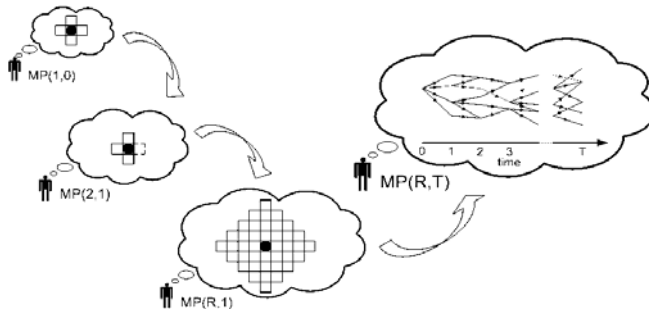


Fig. 2. Expanding neighbourhood of player (R corresponds to the size of the neighbourhood of player's vision in learning processes and T corresponds to the number of time steps accounting in the CA rules; $T=1$ corresponds to the absent of anticipation)

The case of $MP(R, T)$ in upper right angle of picture illustrates important possibilities in CA models with anticipation. The graph illustrates the origin of multivaluedness of the presumable states in the models. It corresponds to the case when each player anticipates existing of many presumable states of elements of the system. Such uncertainty generating by mentality accounting of players is the example of manifestation of 'strong' anticipation. It follows from our investigations that anticipation and multivaluedness also may serve as the source of uncertainty in the systems.

4 Conclusions

In proposed paper we have presented some ways and tools for an improvement the CA based models and their software. The presented results are interesting for practical applications. Described models may be used as the polygon for testing new approaches and ideas in the field of cellular automata investigations. Also some new possibilities are proposed which are connected with the mentality of players.

References

1. <http://www.robocup.org>
2. Dubois, D.: Generation of fractals from incursive automata, digital diffusion and wave equation systems. *BioSystems* 43, 97–114 (1997)
3. Makarenko, A., Goldengorin, B., Krushinski, D.: Game 'Life' with Anticipation Property. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 77–82. Springer, Heidelberg (2008)
4. Goldengorin, S.B., Krushinski, D., Makarenko, A.: Synchronization of Movement for Large – Scale Crowd. In: Kyamakya, K., Halang, W.A., Unger, H., Chedjou, J.C., Rulkov, N.F., Li, Z. (eds.) *Recent Advances in Nonlinear Dynamics and Synchronization: Theory and applications*, pp. 277–303. Springer, Heidelberg (2009)

The Complexity of Three-Dimensional Critical Avalanches

Carolina Mejía and J. Andrés Montoya

Universidad Industrial de Santander, Bucaramanga, Colombia
caromejia@uis.edu.co, juamonto@uis.edu.co

Abstract. In this work we study the complexity of the three-dimensional sandpile avalanches triggered by the addition of two critical configurations. We prove that the algorithmic problem consisting in predicting the evolution of three dimensional critical avalanches is the hardness core of the three-dimensional Abelian Sandpile Model. On the other hand we prove that three-dimensional critical avalanches are superlinear long on average. It suggests that the prediction problem is superlinear-hard on average.

Can we quickly predict the evolution of an avalanche if we are given a full description of the initial conditions? *The Abelian Sandpile Model* has been used to simulate dissipative dynamical process such as forest fires, earth quakes, extinction events, and (off course) avalanches [2]. Can we quickly predict sandpile avalanches? There is some previous work concerning the computational complexity of prediction problems related to The Abelian Sandpile Model (see for example [3], and [4]). Most of those works are focused on the analysis of The Sandpile Prediction Problem, which refers to the computation of relaxations of unstable configurations. In this work we analyze the complexity of predicting the final state of the avalanches triggered by the addition of two critical configurations, (we focus our research on three-dimensional cubic lattices). Those avalanches are called *critical avalanches*. We show that *GC*, the problem consisting in predicting the evolution of three-dimensional critical avalanches, is at least as hard as most of the algorithmic problems related to The Abelian Sandpile Model, that is: we show that *GC* is the hardness core of the predicting tasks related to the model. It is important to remark that our complexity theoretical analysis is based on the notion of *NC*-Turing reducibility. We have chosen to work with this notion because all the algorithmic problems considered in this paper are *Ptime* computable, and because we are interested in analyzing the *polylogarithmic time computability* of those problems.

We believe that the argued *Self-organized Criticality* [1] of The Abelian Sandpile Model is the complexity source of *GC* and its relatives. We show that *GC* is the complexity core of The Abelian Sandpile Model, and we prove that critical avalanches are *superlinear long on average*. It implies that any sequential simulation algorithm computing *GC* has a running time which is *superlinear on average*. Also, we prove that the criticality of the model implies some type of average-case hardness. We wanted to establish some links between the Self-Organized

Criticality of The Abelian Sandpile Model and the algorithmic hardness of the prediction problems related to it, we believe that we have partially fulfilled this goal.

Organization of the work. This work is organized into three sections. In section one we introduce The Abelian Sandpile Model and we review some basic facts concerning this model. In section two we study the statistics of three-dimensional critical avalanches and we compute their expected length. In section three we study the algorithmic hardness of GC , we show that most algorithmic problems related to The Abelian Sandpile Model are NC^2 -Turing reducible to GC .

1 The Three-Dimensional Abelian Sandpile Model

In this section we introduce the basic definitions and some of the basic results concerning The Abelian Sandpile Model.

Given $n \geq 1$, we use the symbol \mathcal{G}_n to denote the *cubic lattice* of order n , whose vertex set is equal to $[n] \times [n] \times [n]$. We use the symbol \mathcal{L}_n to denote the *cubic sandpile lattice* of order n , which is obtained from \mathcal{G}_n by adding to it a special node $*$ called the *sink*. Furthermore, given v a node on the border of \mathcal{G}_n , there are $6 - \deg_{\mathcal{G}_n}(v)$ edges in \mathcal{L}_n connecting v and $*$. We will use the symbol $V(\mathcal{L}_n)^*$ to denote the set $V(\mathcal{L}_n) - \{*\} = V(\mathcal{G}_n)$. Note that given $v \in V(\mathcal{L}_n)^*$ we have that $\deg(v) = 6$.

A *configuration* on \mathcal{L}_n is a function $g : V(\mathcal{L}_n)^* \rightarrow \mathbb{N}$. Given g a configuration on \mathcal{L}_n and given $v \in V(\mathcal{L}_n)^*$ we say that v is *g -stable* if and only if $g(v) \leq 6$. We say that g is an *stable configuration* if and only if for all $v \in V(\mathcal{L}_n)^*$, we have that v is g -stable.

We can attach to any sandpile lattice \mathcal{L}_n a *Graph Automaton* $\mathcal{SP}(\mathcal{L}_n)$ whose underlying graph is \mathcal{L}_n and whose transition rule is the *toppling rule* defined by:

Given $v \in V(\mathcal{L}_n)^*$ such that $g(v) \geq 6$, we have that $g \rightarrow g_v$ is a possible transition, where g_v is the configuration on \mathcal{L}_n defined by

$$g_v(w) = \begin{cases} g(v) - 6, & \text{if } w = v, \\ g(w) + 1, & \text{if } v \text{ is a neighbor of } w \\ g(w), & \text{if } v \text{ is not a neighbor of } w \end{cases} \quad (1)$$

Any transition of $\mathcal{SP}(\mathcal{L}_n)$ is called a *firing* or a *toppling*. So, given g a configuration, the transition $g \rightarrow g_v$ is a firing, and if such transition occurs we say that node v was fired (toppled) or we say that a firing (toppling) at v has occurred. Given \mathcal{L}_n a sandpile lattice and given g an initial configuration, we can choose an unstable node, fire it and obtain a new configuration. A sequence of firings $g_1 \rightarrow g_2 \rightarrow \dots \rightarrow g_m$ is called an *avalanche* of length $m - 1$ with initial configuration g_1 , and we say that it is an avalanche from g_1 to g_m . If g_m is stable we say that g_m is a *stabilization* or a *relaxation* of g_1 . Given g a configuration on \mathcal{L}_n we use the symbol $ST(n, g)$ to denote the set of relaxations of g . Furthermore, given \mathcal{L}_n , g and $A = g \rightarrow g_1 \rightarrow \dots \rightarrow g_m$ an avalanche, the *score vector* of A ,

which we denote with the symbol SC_A , is equal to $(t_v)_{v \in V(\mathcal{L}_n)^*}$, where for any $v \in V(\mathcal{L}_n)^*$ the entry t_v is equal to the number of times node v was fired during the occurrence of A .

Theorem 1 (The fundamental theorem of sandpiles). *Let n be a natural number and let g be a configuration on \mathcal{L}_n , we have:*

1. *Any avalanche beginning in g is finite.*
2. $|\mathcal{ST}(n, g)| = 1$.
3. *Given A and B two maximal avalanches beginning in g , we have that $SC_A = SC_B$.*

A proof of this theorem can be found in [2].

Remark 1. Given $n \geq 1$ we use the symbol $\mathcal{C}(n)$ to denote the set $\mathbb{N}^{V(\mathcal{L}_n)^*}$ which is equal to the set of all the configurations on \mathcal{L}_n . Given $g \in \mathcal{C}(n)$ we use the symbol SC_g to denote the vector SC_A , where A is any maximal avalanche beginning in g .

Let $\mathcal{ST}(n)$ be the set of all the stable configurations on \mathcal{L}_n . We can define a function $st_n : \mathcal{C}(n) \rightarrow \mathcal{ST}(n)$ where $st_n(g)$ is the stabilization of g .

Note that, for any n the function st_n is computable: given g a configuration on \mathcal{L}_n , if one wants to compute $st_n(g)$, one only has to simulate the automaton $\mathcal{SP}(\mathcal{L}_n)$ on input g .

Given \mathcal{L}_n a sandpile lattice and given f_1, f_2 and f_3 three configurations, we have that

$$st_n(f_1 + f_2 + f_3) = st_n(st_n(f_1 + f_2) + f_3). \tag{2}$$

Last equation allow us to associate to any sandpile graph a sandpile monoid. To this end we define a binary operation $\oplus : \mathcal{ST}(n)^2 \rightarrow \mathcal{ST}(n)$ in the following way

$$f \oplus g = st_n(f + g) = st_n(f) \oplus st_n(g). \tag{3}$$

The pair $(st(n), \oplus)$ is a finite commutative monoid. We will use the name *Sandpile Monoid of \mathcal{L}_n* to denote the pair $\mathcal{M}(n) = (\mathcal{ST}(n), \oplus)$. It is known that the *kernel* of a finite commutative monoid is an abelian group [6]. We use the symbol $\mathcal{K}(n)$ to denote the abelian group $(\text{Ker}(\mathcal{M}(n)), \oplus \upharpoonright_{(\text{Ker}(\mathcal{M}(n)))^2})$, which we call *the critical group of \mathcal{L}_n* . The elements of $\mathcal{K}(n)$ will be called *critical configurations*. Intuitively, critical configurations are stable configurations of high complexity, which are very near to be unstable. This point of view is supported by the following theorem [2].

Theorem 2. *Given \mathcal{L}_n a sandpile lattice and given $f \in \mathcal{M}(n)$ we have that f is a critical configuration if and only if there not exists $A \subseteq V(\mathcal{L}_n)^*$ such that for any $u \in A$ the inequality $\text{deg}_A(u) \not\geq f(u)$ holds.*

Remark 2. Given G a graph, given $A \subseteq V(G)^*$ and given $v \in V(G)$ we use the symbol $\text{deg}_A(v)$ to denote the quantity $|\{w \in A : \{w, v\} \in E(G)\}|$.

Remark 3. Given \mathcal{M} a monoid, its kernel is equal to the intersection of the ideals included in \mathcal{M} . It implies that $\text{Ker}(\mathcal{M}(n))$ is an ideal of $\mathcal{M}(n)$ and it implies that given f a configuration and given g a critical configuration, $f \oplus g \in \mathcal{K}(n)$.

2 The Statistics of Three-Dimensional Critical Avalanches

We use the term *critical avalanches* to denote the avalanches triggered by the addition of two critical configurations. In this section we prove that the expected length of three-dimensional critical avalanches is $\Omega(n^4) = \Omega(|\mathcal{L}_n|^{\frac{4}{3}})$.

Given $f, g \in \mathcal{K}(n)$ we will use the symbol $L(f, g)$ to denote the length of the critical avalanches triggered by $f + g$.

Definition 1. We say that w_n is the maximal critical configuration on \mathcal{L}_n if for any $v \in V(\mathcal{L}_n)^*$, $w_n(v) = 5$.

We observe that given $f, g \in \mathcal{M}(n)$ the inequality $L(f, g) \leq L(w_n, w_n)$ holds. Also, we have that $L(w_n, w_n)$ is an upper bound on avalanche length.

Theorem 3. $L(w_n, w_n) \in \Omega(|\mathcal{L}_n|^{\frac{4}{3}})$.

Proof. We prove that there exists a constant C such that, for any $n \geq 2$, we have

$$L(w_n, w_n) \geq Cn^4 \in \Omega(|\mathcal{L}_n|^{\frac{4}{3}}). \tag{4}$$

Given \mathcal{L}_n a sandpile lattice, we use the symbol $\delta(\mathcal{L}_n)$ to denote the set

$$\{w \in V(\mathcal{L}_n)^* : \{*, w\} \in E(\mathcal{L}_n)\}. \tag{5}$$

We use the symbol δ_n to denote the configuration defined by: given $v \in V(\mathcal{L}_n)^*$, $\delta_n(v) = 6 - \deg_{\mathcal{G}_n}(v)$.

Remember that all the avalanches triggered by $2w_n$ have the same length. Fix $n \geq 2$, we want to lowerbound the length of a very specific avalanche triggered by $2w_n$. Given $n \geq 2$, we can identify the sink of \mathcal{L}_{n-2} with $\delta(\mathcal{L}_n)$ the border of \mathcal{L}_n . If we make such an identification, we can think of \mathcal{L}_{n-2} as embedded into \mathcal{L}_n , and we can express the configuration w_n as $w_{n-2} + \delta_n + \gamma_n$, where γ_n is some configuration on \mathcal{L}_n . Note that

$$2w_n = (w_n + \delta_n) + (w_{n-2} + \gamma_n). \tag{6}$$

We know that

$$\begin{aligned} st_n(2w_n) &= st_n(st_n(w_n + \delta_n) + st_n(w_{n-2} + \gamma_n)), \\ st_n(w_n + \delta_n) &= w_n, \\ L(w_n, \delta_n) &= |V(\mathcal{L}_n)^*| = (n)^3. \end{aligned} \tag{7}$$

Thus, we have that there exists a configuration β_n such that we can pass from the configuration $2w_n$ to the configuration $2w_{n-2} + \beta_n$. Furthermore, we have that the partial avalanche carrying us from $2w_n$ to $2w_{n-2} + \beta_n$ has a length equal to n^3 . This partial avalanche (it is not a maximal avalanche) is the first stage of the whole stabilization process. In the second stage we work on the subgraph \mathcal{L}_{n-2}

with the configuration $2w_{n-2}$. We can claim that after $(n - 2)^3$ topplings we can pass from $2w_{n-2}$ to $2w_{n-4} + \beta_{n-1}$. If we continue in this way, going to the core (center) of \mathcal{L}_n , we have to generate $\lfloor \frac{n}{2} \rfloor - 1$ partial avalanches whose lengths are lowerbounded by $n^3, (n - 2)^3, \dots, (n - 2 (\lfloor \frac{n}{2} \rfloor - 2))^3$ and $(n - 2 (\lfloor \frac{n}{2} \rfloor - 1))^3$ (respectively). Therefore, we have that

$$L(w_n, w_n) \geq \left(\sum_{i=0}^{\lfloor \frac{n}{2} \rfloor - 1} (n - 2i)^3 \right) \in \Omega(n^4) \tag{8}$$

Definition 2. Given $f, g \in \mathcal{C}(n)$ we use the symbol $f \leq g$ to indicate that for any $v \in V(\mathcal{L}_n)^*$ the inequality $f(v) \leq g(v)$ holds.

We will prove that critical avalanches are superlinear long on average. First at all we have to remember the notion of accessibility: given $f, g \in \mathcal{C}(n)$ we say that g is *accessible* from f if and only if there exists a configuration $h \geq g$ and there exists a sequence of nodes such that if we begin with f and we topple the nodes in the sequence, (according to the order determined by the sequence), we obtain h . We will use the symbol $f \rightarrow g$ to indicate that g is accessible from f .

Lemma 1. For all $f_1, \dots, f_{70} \in \mathcal{K}(n)$ the configuration $2w_n$ is accessible from $f_1 + \dots + f_{70}$.

Proof. Given $f \in \mathcal{K}(n)$ and given $\{v, w\} \in E(\mathcal{L}_n)$ we have that either $f(w) \geq 0$ or $f(v) \geq 0$ (see reference [2]). Let f_1, \dots, f_7 be seven critical configurations, given $v \in V(\mathcal{L}_n)^*$ we have that either there exists $i \leq 7$ such that $f_i(v) \geq 0$ or for any w neighbor of v and for any $i \leq 7$ we have that $f_i(w) \geq 0$. Suppose that for all $i \leq 7$ we have that $f_i(v) = 0$, in this case we can choose any neighbor of v , say w , and fire it. Also, we can place at least one chip on v , taking care of leaving at least one chip on w . It is clear that if we begin with the configuration $\sum_{i \leq 7} f_i$ we

can choose a sequence of at most $|V(\mathcal{L}_n)^*|$ topplings to obtain a configuration h such that for any $v \in V(\mathcal{L}_n)^*$ the inequality $h(v) \geq 1$ holds. Then, given $f_1, \dots, f_{70} \in \mathcal{K}(n)$ we have that $\sum_{i \leq 70} f_i \rightarrow 2w_n$.

Theorem 4. (Critical configurations generate, with high probability, very long avalanches) Given $n \geq 1$ we have that

$$\Pr_{f, g \in \mathcal{K}(n)} \left[L(f, g) \geq \frac{L(w_n, w_n)}{2^{70}} \right] \geq \frac{1}{69}. \tag{9}$$

Proof. Given $f_1, f_2, \dots, f_{70} \in \mathcal{K}(n)$ we have that $\sum_{i \leq 70} f_i \rightarrow 2w_n$. It implies that

$$L \left(f_{70}, \sum_{i \leq 69} f_i \right) \geq L(w_n, w_n). \tag{10}$$

Also, we have that either

$$\left(L \left(f_{70}, \bigoplus_{i \leq 69} f_i \right) \geq \frac{L(w_n, w_n)}{2} \right) \text{ or } \left(L \left(f_{69}, \sum_{i \leq 68} f_i \right) \geq \frac{L(w_n, w_n)}{2} \right). \quad (11)$$

Arguing in this way we can prove that there exists $i \leq 70$ such that

$$L \left(f_i, \bigoplus_{j \leq i-1} f_j \right) \geq \frac{L(w_n, w_n)}{2^{70}}. \quad (12)$$

Thus, we have that

$$\Pr_{f_1, \dots, f_{70}} \left[\exists i, i \leq 70 \left(L \left(f_i, \bigoplus_{j \leq i-1} f_j \right) \geq \frac{L(w_n, w_n)}{2^{70}} \right) \right] = 1. \quad (13)$$

Note that for any $f \in \mathcal{K}(n)$ and for any $i \geq 1$ we have that

$$\Pr_{f_1, \dots, f_i} \left[\bigoplus_{j \leq i} f_j = f \right] = \frac{1}{|\mathcal{K}(n)|}. \quad (14)$$

Given f_1, \dots, f_α a sequence of critical configurations on \mathcal{L}_n and given $i \leq \alpha - 1$, we define $g_i = \bigoplus_{j \leq i} f_j$. We have that:

1. The procedure below is a sound method to generate, uniformly at random, two elements of $\mathcal{K}(n)$.
 - Choose uniformly at random f_1, \dots, f_α , ($\alpha \geq 2$).
 - Choose uniformly at random $i \in \{2, \dots, \alpha\}$.
 - Compute f_i and g_{i-1} .
2. It holds that

$$\Pr_{f_1, \dots, f_{70}} \left[\exists i, 2 \leq i \leq 70 \left(L(f_i, g_{i-1}) \geq \frac{L(w_n, w_n)}{2^{70}} \right) \right] = 1. \quad (15)$$

From items 1 and 2 we obtain

$$\begin{aligned} & \Pr_{f, g \in \mathcal{K}(n)} \left[L(f, g) \geq \frac{L(w_n, w_n)}{2^{70}} \right] = \\ & \Pr_{2 \leq i \leq 70; f_1, \dots, f_{70}} \left[L(f_i, g_{i-1}) \geq \frac{L(w_n, w_n)}{2^{70}} \right] \geq \frac{1}{69} \end{aligned} \quad (16)$$

Thus, we have proven that

$$\Pr_{f, g \in \mathcal{K}(n)} \left[L(f, g) \geq \frac{L(w_n, w_n)}{2^{70}} \right] \geq \frac{1}{69}. \quad (17)$$

Let $X_n : \mathcal{K}(n)^2 \rightarrow \mathbb{N}$ be the random variable defined by $X_n(f, g) = L(f, g)$.

Theorem 5. $E[X_n]$, the expected value of X_n , belongs to $\Omega(n^4)$.

Proof. We know that there exists a positive constant K such that

$$\Pr_{f,g \in \mathcal{K}(n)} [X_n(f, g) \geq Kn^4] \geq \frac{1}{69}. \tag{18}$$

Then, we have that

$$\frac{K}{69}n^4 \leq E[X_n]. \tag{19}$$

Therefore, we have that $E[X_n] \in \Omega(n^4) = \Omega(|\mathcal{L}_n|^{\frac{4}{3}})$.

2.1 The Algorithmic Hardness of GC

In this section we prove that the addition of critical configurations is, in a very specific sense, the complexity source of The Abelian Sandpile Model. Let $n \geq 1$, it is known that if we simulate the dynamics of the model on \mathcal{L}_n , alternating the adding of fresh chips with the relaxation process, we will arrive after a polynomial number of iterations to the set of critical (also called *recurrent*) configurations. Furthermore, once we enter $\mathcal{K}(n)$ we can not exit this set. It is the case since $\mathcal{K}(n)$ is the stationary state of The Abelian Sandpile Model on \mathcal{L}_n [2]. Also, if we want to efficiently simulate the dynamics of the model we have to be able to compute the addition of any pair (f, g) of configurations, where f is critical and g is stable. We introduce a related problem below, which we denote with the symbol MC^* , and we prove that MC^* is NC^2 Turing reducible to GC .

The Sandpile Prediction Problem, is the algorithmic problem defined by:

Problem 1. (*SPP, sandpile prediction*)

- *Input:* (n, g) , where $n \in \mathbb{N}$ and $g \in \mathcal{C}(n)$.
- *Problem:* Compute $st_n(g)$.

Remark 4. Tardos' bound [5] implies that *SPP*, and each one of the algorithmic problems introduced below, can be solved in polynomial time, because of this we will analyze the relative complexity of those problems using the notion of NC -Turing reducibility.

A Second problem is MC , which corresponds to the computation of the monoid operation \oplus .

Problem 2. (*MC, monoid computations*)

- *Input:* (n, f, g) , where $n \in \mathbb{N}$ and $f, g \in \mathcal{M}(n)$.
- *Problem:* Compute $f \oplus g$.

Now, we introduce the problem GC which is the restriction of *SPP* to critical avalanches.

Problem 3. (GC, group computations)

- *Input:* (n, f, g) , where $n \in \mathbb{N}$ and $f, g \in \mathcal{K}(n)$.
- *Problem:* Compute $f \oplus g$.

Let us introduce three additional problems, which will be play an important role in our analysis.

Problem 4. (SC, computation of score vectors)

- *Input:* (n, f) , where $n \in \mathbb{N}$ and $f \in \mathcal{C}(n)$.
- *Problem:* Compute the vector SC_f .

Problem 5. (MC, mixed computations)*

- *Input:* (n, f, g) , where $n \in \mathbb{N}$, $f \in \mathcal{K}(n)$ and $g \in \mathcal{M}(n)$.
- *Problem:* Compute $f \oplus g$.

Given \mathcal{L}_n a three-dimensional sandpile lattice, we use the symbol $e_{\mathcal{K}(n)}$ to denote the identity of $\mathcal{K}(n)$.

Lemma 2. *Identities can be computed in constant time, if oracle access to GC is provided.*

Proof. In order to compute the identity of $\mathcal{K}(n)$, in constant time and using an oracle for GC, we can use the equations:

1. $w_n^{-1} = w_n - (w_n \oplus w_n)$.
2. $e_{\mathcal{K}(n)} = w_n \oplus w_n^{-1}$.

Lemma 3. *Inverses can be computed in time $O(\log(n))$ if oracle access to GC is provided.*

Proof. Let $v \in V(\mathcal{L}_n)^*$ and let $w_v = w_n - e_v$. It follows from theorem 2 that w_v is a critical configuration. Let $f \in \mathcal{K}(n)$, note that

$$f^{-1} = \left(\bigoplus_{v \in V(\mathcal{L}_n)^*} f(v) w_v \right) \oplus \underbrace{\left(w_n^{-1} \oplus \dots \oplus w_n^{-1} \right)}_{\|f\| \text{ times}}. \quad (20)$$

It is clear that we can compute the expression on the right side of the equation above in time $O(\log(n))$ and using a polynomial number of processors, (if oracle access to GC is provided).

Given v an element of $V(\mathcal{L}_n)^*$, we use the symbol e_v to denote the configuration

$$e_v(w) = \begin{cases} 1, & \text{if } v = w \\ 0, & \text{otherwise} \end{cases} \quad (21)$$

Let $e_n : V(\mathcal{L}_n)^* \rightarrow \mathcal{K}(n)$ be the function defined by $e_n(v) = e_{\mathcal{K}(n)} \oplus e_v$ and let $e : \mathbb{N}^3 \times \mathbb{N} \rightarrow \left(\bigcup_{i \geq 1} \mathcal{K}(n) \right) \cup \{\infty\}$ be the function defined by

$$e(v, n) = \begin{cases} e_n(v), & \text{if } v \in V(\mathcal{L}_n)^* \\ \infty, & \text{else} \end{cases}. \tag{22}$$

Problem 6. (EC, computation of e)

- *Input:* (n, v) , where $n \in \mathbb{N}$ and $v \in V(\mathcal{L}_n)^*$.
- *Problem:* Compute $e(v)$.

Next theorem is the main theorem of this section.

Theorem 6. *(The relative hardness of sandpile prediction problems)*

1. *SPP and SC are NC^2 -Turing equivalent.*
2. *SPP is NC^2 -reducible to MC.*
3. *MC* can be computed in time $O(\log^2(n))$ if oracle access to EC and GC is provided.*
4. *EC is NC-Turing reducible to GC.*
5. *The problems MC* and GC are NC^2 -Turing equivalent.*

Proof. The proof of item 1 can be found in [3]. The proof of item 2 is very easy, also we prove items 3 and 4, item 5 follows from items 3 and 4.

1. (Proof of item 3) Let (n, f, g) be an instance of MC^* . We observe that

$$f \oplus g = f \oplus g \oplus \underbrace{e_{\mathcal{K}(n)} \oplus \dots \oplus e_{\mathcal{K}(n)}}_{\|g\|\text{-times}}. \tag{23}$$

If we express g as $\sum_{v \in V(\mathcal{L}_n)^*} m_v e_v$ we get

$$f \oplus g = f \oplus \left(\bigoplus_{v \in V(\mathcal{L}_n)^*} m_v e_n(v) \right). \tag{24}$$

Also, we can use n^3 processors to compute $\{m_v e_n(v)\}_{v \in V(\mathcal{L}_n)^*}$, this computation takes $O(\log^2(n + \|g\|))$ time units, since we are supposing that we have oracle access to EC . We can use the same n^3 processors to compute

$f \oplus \left(\bigoplus_{v \in V(\mathcal{L}_n)^*} m_v e_n(v) \right)$ in time $O(\log^2(n + \|f\| + \|g\|))$, since we are supposing that we have oracle access to GC .

2. (Proof of item 4) Observe that

$$e_n(v) = e_v \oplus e_{\mathcal{K}(n)} = e_v \oplus (w_v \oplus w_v^{-1}) = w_n \oplus w_v^{-1}. \tag{25}$$

Thus, if one wants to compute $e_n(v)$, one only has to compute $w_n \oplus w_v^{-1}$ (note that $w_n, w_v^{-1} \in \mathcal{K}(n)$). We can compute w_v^{-1} in time $O(\log(n))$ if oracle access to GC is provided. Then, we can solve EC in time $O(\log(n))$ using an oracle for GC .

Next theorem follows easily from the results obtained in section 2, it brings together the results concerning the algorithmic hardness of GC and the results concerning the statistics of critical avalanches. Let \mathcal{SA} be the naive (sequential) sandpile automata simulation algorithm, and let \mathcal{B} be the parallel sandpile automata simulation algorithm (we topple all the unstable nodes at once). We will use the symbol $t_{\mathcal{SA}}(n, f, g)$ to denote the running time of \mathcal{SA} on input (n, f, g) , (we define $t_{\mathcal{B}}(n, f, g)$ accordingly).

Theorem 7. *Let $n \geq 1$ be a natural number*

1. *There exists a positive constant K such that*

$$\Pr_{f,g \in \mathcal{K}(n)} [t_{\mathcal{SA}}(n, f, g) \geq Kn^4] \geq \frac{1}{69}. \quad (26)$$

2. *There exists a positive constant R such that*

$$\Pr_{f,g \in \mathcal{K}(n)} [t_{\mathcal{B}}(n, f, g) \geq Rn] \geq \frac{1}{69}. \quad (27)$$

Theorem 7 suggests that the problem GC is $n^{\frac{1}{3}}$ -hard on average, which means that given an algorithm \mathcal{M} computing the problem GC , there exists two positive constants K, D such that

$$\Pr_{f,g \in \mathcal{K}(n)} [t_{\mathcal{M}}(n, f, g) \geq Kn] \geq D. \quad (28)$$

Let us finish this work stating the following conjecture.

Conjecture 1. The problem GC is $n^{\frac{1}{3}}$ -hard on average.

Acknowledgement. Thanks to VIE-UIS and thanks to Colciencias research project 111518925292.

References

1. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized Criticality. *Physical Review A* 38, 364–374 (1988)
2. Dhar, D.: Theoretical Studies of Self-organized Criticality. *Physica A* 369, 29–70 (2006)
3. Mejía, C., Montoya, A.: On the Algorithmic Complexity of the Abelian Sandpile Model. In: *Proceedings of Automata 2009*, pp. 147–162 (2009) (submitted)
4. Moore, C., Nilsson, M.: The computational complexity of sandpiles. *Journal of Statistical Physics* 96, 205–224 (1999)
5. Tardos, G.: Polynomial bound for a chip firing game on graphs. *SIAM J. Discrete Mathematics* 1, 397–398 (1988)
6. Toumpakari, E.: On the abelian sandpile model. Ph.D. Thesis, University of Chicago (2005)

Using Cellular Automata on a Graph to Model the Exchanges of Cash and Goods

Ranaivo Mahaleo Razakanirina and Bastien Chopard

University of Geneva, Switzerland
ranaivo.razakanirina@unige.ch,
bastien.chopard@cui.unige.ch

Abstract. This paper investigates the behaviors and the properties of a “Give and Take” cellular automaton on a graph. Using an economical metaphor, this model implements the exchange of cash against goods, among the nodes of a graph G , with a local pricing mechanism. During the time evolution of this model, the strongly connected components (SCC) emerge, mimicking the creation of independent sub-markets. In the steady state, each SCC is characterized by a unique price obeying the supply and demand law for that sub-market. We also show that the distributions of cash and goods are proportional to the indegree of the cells, reproducing a Zipf’s law of wealth distribution in case of a scale-free graph topology.

Keywords: Complex system, cellular automata on a graph, complex network, strongly connected components, economical model.

1 Introduction

A complex system is an organization which consists of many parts and the interaction between them [11, 13]. This approach gains in popularity and offers a new way to the scientists and the researchers to model complex phenomena in various real world applications. Complex phenomena can hardly be solved analytically by mathematical models and numerical simulations are needed. Complex networks [6] are now widely used to describe interaction patterns in social or economical systems. We can cite among other applications the evolution of the structure of complex networks [4] or the modeling the propagation of economic crises [14].

Cellular automata [7] are an effective tool to study complex systems and it is natural to consider their extension to a graph topology. In [8, 13] we have defined cellular Automata on Graph (CAG), a formalism that extends the power of classic cellular automata approach by introducing irregularity and dynamics on the neighborhood relationship.

To illustrate the CAG approach, we consider here a particular case of a CAG which we called the “Give and Take” model (GT model), which implements a simple economical interaction between agents. We show that this model, interpreted as a market dynamics, produces interesting results: spontaneous

sub-market creation, emergence of a global price in each sub-market, compliance with the supply-demand law and Zipf's type of wealth distribution.

The paper is organized as follows: we first define formally the GT-model, then we discuss its computer implementation on the CAG engine. Finally, we present the simulation results and formulate mathematically the properties of the model.

2 GT (Give and Take) Model Formalism

The GT-model simulates the complex interactions between agents exchanging goods against cash. The model is built with a directed graph G which has n cells and m edges. One cell corresponds to one agent. A directed edge (i, j) of the graph G (see Fig. 1) models the buying and selling actions between i and j : agent i gives cash to buy goods from agent j and, in exchange, j returns a certain quantity of goods.

Cells (or agents) can have both the role of a buyer or a seller. As a buyer, a cell distributes its cash to each seller it is connected to, in a way which is inversely proportional to the price offered by the seller. In turns, each seller distributes its good in proportion to the money received from each buyer. For each buyer-seller relation, a new price is computed, as the amount of money spent over the amount good received.

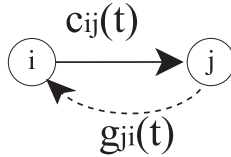


Fig. 1. The flows of cash used to buy goods are indicated by the direction of the edges of the graph G . Here, the flow comes from i to j . The dotted edge represents the direction of the flow of goods in exchange, which is the opposite direction of the flow of cash.

Table 1 summarizes all the notations used to describe the GT-model.

The state of each cell i at iteration t consists of the cash amount $c_i(t)$ and the quantity of goods $g_i(t)$ owned by the cell. We assume that these quantities are infinitely divisible. To prevent the rainy days, at each iteration, each cell invests only a fraction λ_i of its cash and a fraction μ_i of its goods. The amount of cash offered from agent i to agent j is denoted by $c_{ij}(t)$ and the amount of goods given in exchange by g_{ji} . The price proposed by seller j to buyer i is denoted as $p_{ij}(t)$.

The dynamic of GT-model at each time step t is composed of four phases, treated successively as follow:

Table 1. Summary of notation

Symbol	Meaning
$c_i(t)$	Cash owned by cell i at the time iteration t
$g_i(t)$	The quantity of goods owned by the cell i at the time iteration t
λ_i	The fraction of cash invested by the cell i to others at each iteration. $0 \leq \lambda_i \leq 1$
μ_i	The fraction of goods invested by the cell i to others at each iteration. $0 \leq \mu_i \leq 1$
c_{tot}	The total cash in the whole CAG
g_{tot}	The total amount of goods in the whole CAG
$c_{ij}(t)$	The flow of cash from i to j at the time iteration t
$g_{ji}(t)$	The flow of goods from j to i at the time iteration t
$p_{ij}(t)$	The unit price of goods proposed by j at the time iteration t
k_i^{in}	The indegree of the cell i
k_i^{out}	The outdegree of the cell i
N_i^{in}	The set of the incoming neighbors of i or the buyers connected with i
N_i^{out}	The set of the outgoing neighbors of i or the sellers connected with i

- *Giving phase:* During this phase, each cell gives cash to its connected sellers. The buyers obey the following strategy: “give more cash to the sellers that propose a better price”. This cash value is inversely proportional to the unit price of goods proposed by the sellers. Mathematically, the resulting cash flow can be expressed as [13]

$$c_{ij}(t) = \frac{p_{ij}^{-1}}{\sum_{l \in N_i^{out}} p_{il}^{-1}(t)} \lambda_i c_i(t), \quad j \in N_i^{out}. \quad (1)$$

At $t = 0$, the initial price can be defined randomly, or assumed to be equal for each seller.

- *Taking phase:* In exchange, each cell returns a certain quantity of goods to its buyers. The highest bidder wins the highest amount of goods. This quantity of goods is proportional to the cash received at the giving phase. Thus the the flow of goods is given by [13],

$$g_{ji}(t) = \frac{c_{ij}(t)}{\sum_{l \in N_j^{in}} c_{lj}(t)} \mu_j g_j(t) \quad i \in N_j^{in}. \quad (2)$$

- *Self-adapting price:* The unit price of goods at the next iteration $t + 1$ is the ratio between the cash given at the giving phase and the quantity of goods in exchange at the taking phase. From (2) we have

$$p_{ij}(t + 1) = \frac{c_{ij}(t)}{g_{ji}(t)} = \frac{\sum_{l \in N_j^{in}} c_{lj}(t)}{\mu_j g_j(t)} \equiv p_j(t + 1). \quad (3)$$

We see in (3) that the unit price of goods actually depends only on the seller j . In other words, the seller proposes the same unit price of goods to all its

connected buyers. For this reason we can simplify the notation and write $p_j(t)$ instead of $p_{ij}(t)$.

- *Edge dynamic:* When a buyer is connected to several sellers, it may decide to stop interacting with one of them, if the offered price is too high in comparison with the others. A buyer decides also to stop the transaction with one seller if the quantity of goods offered is too low. Formally

$$(i, j) \begin{cases} \text{cut} & \text{if } p_j(t) > \tau \min(p_l(t)) \text{ or } g_{ji}(t) < \epsilon, \quad l \in N_i^{out} \\ \text{not cut,} & \text{otherwise} \end{cases} \quad (4)$$

where τ and ϵ are a parameters.

Then, the state of each cell i at the next iteration $t + 1$ is given by the balance of cash and goods,

$$c_i(t + 1) = c_i(t) - \sum_{j \in N_i^{out}} c_{ij}(t) + \sum_{k \in N_i^{in}} c_{ki}(t) \quad (5)$$

$$g_i(t + 1) = g_i(t) - \sum_{j \in N_i^{out}} g_{ij}(t) + \sum_{k \in N_i^{in}} g_{ki}(t) \quad (6)$$

3 GT-Model Simulator and Simulations Setup

The GT-model simulator is the tool that allows the user to interact with the settings and the evolution of the GT-model and to visualize and save the results. It is based on the general CAG formalism described in [8]. This tool, illustrated in Fig. 2, is composed of the following modules:

- **CAG Evolution:** This is the core of the architecture. It implements the dynamics of the GT-model.

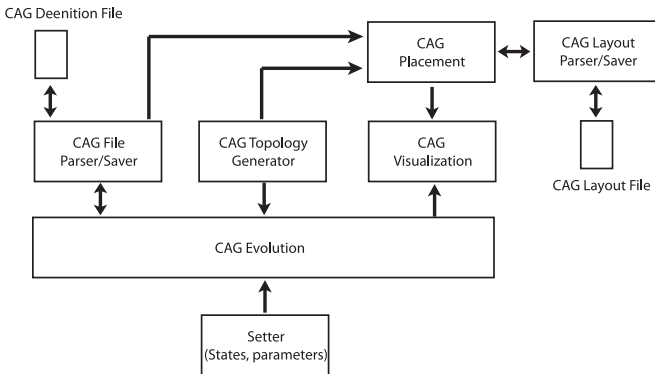


Fig. 2. The architecture of the GT-model simulator

- **Setter:** This module is in charge of initializing the state of each cell, the unit price of goods of each seller, the flow of cash and goods through the edges at any time iteration t .

The initial conditions of the evolution are set by this module. We use two initial conditions which are EQ and RND initial conditions. The total amount of cash and goods in the whole GT-model are divided equally to the cells with EQ initial conditions and on the other hands, divided randomly with RND initial conditions. The initial unit prices of goods are either equals or randoms or chosen by the user.

We choose the same value of λ and the same value of μ for all the cells.

- **CAG File and CAG Layout Parser/Saver:** The CAG Definition File contains all the information about the states of each cell, the flows transiting through each edge. The CAG Layout File contains all the (X, Y) coordinates of each cell on the visualization screen. These Parser/Saver modules parse and convert the Definition and Layout files to the data structure used by the CAG Evolution module. Vice versa they save the snapshot of the evolution at given time iteration to the Definition and Layout files.
- **CAG Topology Generator:** This module generates GT-model based on regular, random and scale-free-graph.
- **CAG Placement and CAG Visualization:** Before the visualization, the 2D coordinates of each cell are calculated by the CAG Placement module using a Force Directed Placement algorithm. Color scale are used by the CAG visualization module to visualize the state of each cell and the fluxes of cash and goods through each edge. Red color for the highest value, orange for the middle and green for the lowest value.

In addition to these modules, our implementation allows the user to modify the data of the model at run time. For instance a new link can be created, or the amount of cash or goods can be modified in a chosen cell.

We have performed experiments using the EQ and RND initial conditions on the following graph topologies: Erdős-Rényi random graph [9,13] (symmetric or not), scale-free graph [12,11,2,13] (symmetric or not), strongly connected graph (SCG) [10]. The values we choose for the parameters are $\lambda = 0.2$, $\mu = 0.3$, $\epsilon = 0.01$ and $\tau = 10$.

4 Results and Discussions

The evolution of the GT-model can be first described by a transient regime in which prices, goods and cash flows are time dependent and links can be cut. During this phase, it is observed that the cutting rule has the effect of make the Strongly Connected Components (SCC) of the graph emerge. The SCC [10] are the sub-graphs such that there is a return path between any pairs of nodes in the SCC. In other words, the GT-model is such that a link in G survives only if all the money or goods that flow through this edge has a path back to where it came from, even if this path is long. Figure 3 depicts the states of `erd_n30_m39_eq`

(a)

(b)

Fig. 3. Evolution of the `erd_n30_m39_eq` GT-model, during its transient regime. This topology is constructed by Erdős-Rényi algorithm with $n = 30$, $m = 39$ and simulated with EQ initial conditions (a) at $t = 0$ and (b) at $t = 100$.

GT-model at $t = 0$ and $t = 100$. We observe clearly at $t = 100$ the emergence of the following SCC: $\{1,18,28\}$, $\{14,19,23\}$, $\{4,6,29,27,21\}$.

Using the economical interpretation, each emerging SCC corresponds to an independent submarket in which the total amount of cash and goods is constant. When reaching the stationary state the unit price of goods inside each SCC is observed to converge to a uniform value. Let us denote this equilibrium price as p_e . The fluxes of cash and goods transiting through each cell are also in equilibrium. Each cell gives to its sellers the exact amount of cash received from its buyers

$$\lambda_i c_i = \sum_{j \in N_i^{in}} c_{ji} \quad (7)$$

Therefore, from (3) we have

$$p_e = \frac{\sum_{j \in N_i^{in}} c_{ji}}{\mu_i g_i} = \frac{\lambda_i c_i}{\mu_i g_i} . \quad (8)$$

Since p_e is observed to be independent of i in the steady state, we have $\mu_i g_i = p_e \lambda_i c_i$ for all i in the same sub-market. When all the cells make the same investment of cash and goods ($\lambda_i = \lambda$ and $\mu_i = \mu$), we can sum this relation over i and we obtain that the equilibrium price is the ratio between the total investment of cash and the total investment of goods

$$p_e = \frac{\lambda C_{tot}}{\mu g_{tot}} . \quad (9)$$

In other words the price is determined by a global supply (μg_{tot}) and demand (λC_{tot}) law.

We can write a mathematical equation for the distribution of cash and goods in a sub-market. We introduce a_{ij} , the element of the adjacency matrix A of the graph G and k_i^{out} the outdegree of the cell i). From the stationary assumption we have

$$\lambda_i c_i = \sum_{j \in N_i^{in}} c_{ji} = \sum_{j \in N_i^{in}} a_{ji} c_{ji} \quad (10)$$

and, since the price is uniform, we have from (11)

$$c_{ij} = \frac{\lambda_i c_i}{k_i^{out}} = \frac{\lambda_i c_i}{\sum_{\ell \in N_i^{out}} a_{i\ell}} \quad (11)$$

Therefore the values c_i obey

$$\lambda_i c_i = \sum_{j \in N_i^{in}} \frac{a_{ji}}{\sum_{\ell \in N_j^{out}} a_{j\ell}} \lambda_j c_j \quad (12)$$

Using in addition that $c_{tot} = \sum_i c_i$, we can solve analytically (12) for a symmetric graph ($a_{ij} = a_{ji}$). We find (13)

$$c_i = \frac{k_i^{in}}{m} c_{tot} . \quad (13)$$

Thus the amount of cash owned by each cell is proportional to the number of its connected buyers and inversely proportional to the market size defined by the number of edges m of the graph G . In a scale-free graph, the node degrees obey a power law distribution and the above relation shows that the wealth distribution follow a Zipf's law.

Compared with the results obtained during the simulations, the analytical results fit perfectly well the simulation results, as illustrated in Fig. 4 in case of symmetric graph G .

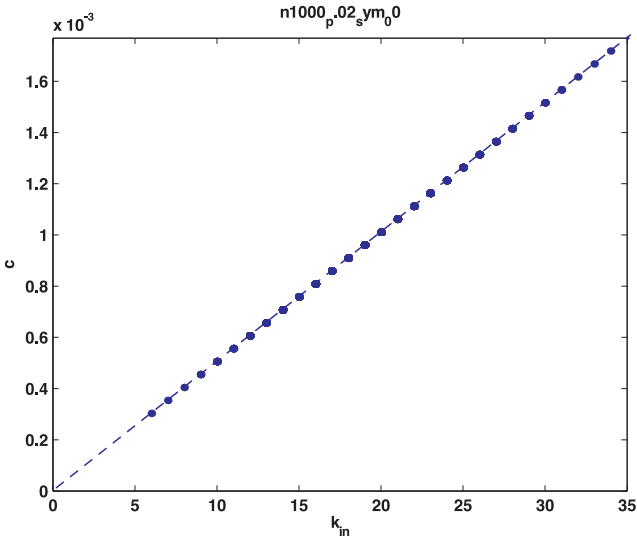


Fig. 4. The dots show the distribution of cash of a stationary GT-model based on symmetric random graph with $n = 1000$ and $m = 19790$ versus indegree k^{in} . The dotted line is the distribution of cash calculated analytically.

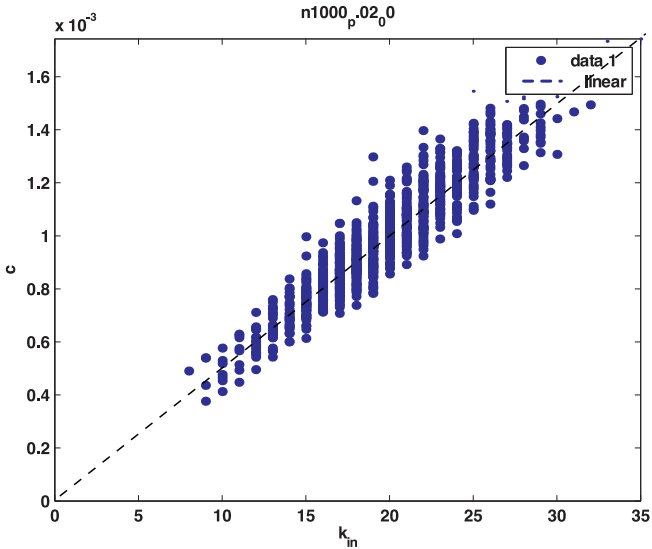


Fig. 5. The dots show the distribution of cash of a stationary GT-model based on asymmetric random graph with $n = 1000$ and $m = 19800$ (a) versus indegree k^{in} . The dotted line is the linear estimation of the distribution given by Eq. (13).

In the case of an asymmetric graph, one has to solve numerically (12). This is easily done for any given adjacency matrix, as (12) is a linear system of equation for c_i . Figure 5 shows the values of the fortune as a function of the indegree of the graph. We see that nodes with the same degree may have different amount of cash. However, we observe that these values are globally compatible with the prediction of (13). We believe that as the number of nodes increases the dispersion will reduce.

5 Conclusion and Future Work

In this work, we have studied the behavior of a particular cellular automaton on a graph (CAG) called Give and Take model (GT-model). This model simulates the exchange of cash against goods between the cells. At each time iteration t , each cell i gives cash to its outgoing neighbors j and in exchange takes goods from them. This relation is represented by an edge (i, j) of the graph G of the CAG. The graph topology also evolves: relations that become too expensive are progressively abandoned.

We found that during the transient regime, the strongly connected components of the graph emerge and form independent sub-markets.

In the stationary regime, and within each sub-market we observed that the unit price of goods becomes homogeneous and obeys a supply and demand law. By the simulations and analytic calculations, the amount of cash owned by each cell is proportional to the number of its connected buyers and inversely proportional to the size of the “market”. Thus, the distribution of cash and goods in the whole automata depends only on the topology. For scale-free graph a power law for the wealth distribution is then observed.

From an application point of view, we plan to extend the model by allowing the production of goods and money (open systems) and by adding a work-salary market.

From the computer science point of view, we plan to analyze the performance of our model as an algorithm to detect the SCC. Further work are in progress to parallelize the CAG evolution algorithm and to standardize the simulator to allow the definition of more CAG models.

References

1. Albert, R., Barabási, A.-L.: Topology of evolving networks: local events and universality. *Physical Review Letters*, 5234 (2000)
2. Barabási, A.-L., Albert, R., Jeong, H.: Mean-Field Theory for Scale-Free Random Networks. *Physica A*, 173–187 (1999)
3. Bar-Yam, Y.: *Dynamics of Complex Systems* (2003)
4. Braha, D., Bar-Yam, Y.: *The Statistical Mechanics of Complex Product Development: Empirical and Analytical Results*. *Microeconomics* (2005)
5. Berge, C.: *Graphs and hypergraphs* (1976)
6. Caldarelli, G.: *Scale-Free Networks*. Oxford University Press, Oxford (2007)

7. Chopard, B., Droz, M.: Cellular Automata modeling of physical systems. Cambridge University Press, Cambridge (1998)
8. Chopard, B., Falcone, J.-L., Razakanirina, R.M., Hoekstra, A., Caiazzo, A.: On the Collision-Propagation and Gather-Update Formulations of a Cellular Automata Rule. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) ACRI 2008. LNCS, vol. 5191, pp. 144–151. Springer, Heidelberg (2008)
9. Durrett, R.: Random Graph Dynamics (2006)
10. Hartmann, A.K., Weigt, M.: Phase Transitions in Combinatorial Optimization Problems (2005)
11. Mitchell, M., Newman, M.: Complex Systems Theory and Evolution. In: Encyclopedia of Evolution (2002)
12. Li, L., Alderson, D., Tanaka, R., Doyle, J.C., Willinger, W.: Towards a Theory of Scale-Free Graphs: Definition, Properties, and Implications (Extended Version). In: Internet Mathematics (2005)
13. Razakanirina, R.M.: Cellular automata over graph applied on financial and goods flows simulation. Master's thesis, University of Geneva, Computer Science department, Master thesis (2007)
14. Angeles Serrano, M., Boguna, M., Vespignani, A.: Patterns of dominant flows in the world trade web. In: Quantitative Finance Papers (2007)

Montebello: A Metapopulation Based Model of Carcinogenesis

David Tuck¹, Willard Miranker², and Jose Costa¹

¹ Department of Pathology, ² Department of Mathematics,
Yale University, New Haven, CT, USA
{david.tuck, willard.miranker, jose.costa}@yale.edu

Abstract. The dynamics of the lengthy process by which tumors arise from normal tissues is not well understood. We developed a stochastic cellular automaton model based on the ecological concept of metapopulations to explore the role of mutation, exogenous disturbance, and selection in the genesis of tumors. The operation of the model shows how disturbances (e.g. inflammation) acting on tissues can cause tumors by modifying the dynamics among metapopulations of cells. Simulations demonstrate that disturbance, without change in mutation rates, can drive tumor formation. Changes in the distribution of genetic alterations among metapopulations in the tissue can predict the emergence of a tumor, thus providing a measure of risk. Modifying the disturbance regimen can prevent the emergence of tumors. Thus, the model provides insights into how mutation rates and disturbance interact in the causation of cancer, and illustrate how measuring metapopulation distributions can provide surrogate end points for preventive intervention.

1 Introduction

Evolutionary concepts and ecological theory have been applied to the study of cancer and have contributed to the generation of new hypotheses [1-6]. During the process of carcinogenesis, the emergence of a malignant phenotype depends on a series of factors, some of which have a strong effect on diversity (e.g., mutation rate, niche size). Disturbance (any exogenous cause of death) is a powerful agent altering the dynamics among populations, influencing both their stability and diversity. At low levels of disturbance, competitively dominant taxonomic species exclude subordinate species and excessive disturbance leads to local extinction. Intermediate levels of disturbance balance these two poles and maximize diversity [7-9].

2 The Model

The dynamic constitution of a tissue is simulated by a 200x200 cellular automaton. Each cell on the grid represents a microenvironment (patch) which may be occupied by one or more clones which can be quiescent, expand to neighboring patches or die. Each patch has a basal rate of division as well as senescence and death for each clone constituting the patch. Each cell is endowed with 2^{10} genes. Mutations in one gene

can occur at 10 different alleles, all leading to an altered phenotype. This phenotype is characterized by an increased probability of a patch to expand into an empty neighboring space (proliferative potential); increased probability of cellular survival at each time step (e.g., an apoptotic defect), or a state of altered susceptibility to exogenous disturbances. Mutations in the great majority of genes, reflecting the deleterious effect of accumulating mutations, cause an increased probability of cellular death. This scenario simulates the metapopulation dynamics of a healthy undisturbed tissue under background mutational rates. The relative frequencies of 30 alleles are tracked to record the variational change resulting from the evolution among the cell populations constituting the tissue. The steady dynamic risk-free state can be altered by disturbances that randomly kill patch populations. The disturbance regimen is specified by the interval between disturbances and the intensity of the disturbance (probability of death of an affected patch). We simulate global disturbances which have an equal probability of affecting all cells in the model. Disturbance simulates exogenous pathologies, such as repeated trauma, cell toxicity or cytolytic infection, which recurrently produce tissue injury due to cell death. Repeated cell loss introduces proliferative pressure within the patch and among neighboring patches as the loss triggers the homeostatic mechanism of tissue repair which is simulated by the local rule of expansion in to neighboring empty space. (The formal description and operation of the Montebello model is given in the supplementary material at <http://genecube.med.yale.edu:8080/montebello>).

3 Results

We first identify model parameters of growth, senescence and death, mutation rates so that in the absence of disturbance there are only minimal deviations from steady state

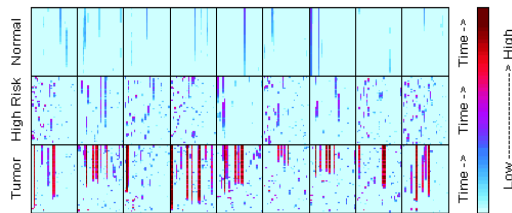


Fig. 1. The totals of each mutant allele are plotted over the full time course (5000 time steps, or possible cell divisions) for twenty-seven selected examples of the simulations. Thirty mutant alleles (the proliferation, death, and susceptibility mutants in order) are aligned along the x axis. A column in each plot shows the total mutant level for that allele over the time course with the baseline time point at the bottom and the final time point at the top of each plot. The top row shows normal (undisturbed) samples in which low levels of individual mutant alleles arise, persist at low levels or die out without expansion. The middle row shows samples exposed to disturbance in which mutations arise frequently but never develop into tumors. Disturbances cause expansion of clones that harbor one or multiple fitness increasing mutations. Most of these clonal populations will collapse before realizing the complex genotype that defines a tumor. The bottom row shows individual simulations in which tumors do form following disturbance. Values for other parameters are provided in the Supplementary Data.

tissue density and very low ($< 1\%$) tumor formation rate with mutation rates consistent with clinical observation in humans [10]. In this risk-free normative state, mutated clones with the potential to progress to tumor arise regularly, persist for various periods but rarely expand (Figure 1a). Under these conditions, the time span in which empty patch-space persists is relatively short-lived. This fits an intuitive conception of structured tissue with tissue repair functions intact. In the absence of disturbance, tumors arise only if the mutation rate is escalated to unrealistic levels.

The introduction of disturbance changes the composition of the tissue. Disturbances cause expansion of clones that harbor one or multiple fitness increasing mutations. Under low levels of disturbance most of these clonal populations will collapse before realizing the complex genotype that defines a tumor (Figure 1b) and be replaced by wild type or patches with a simpler combination of genetic alterations. Samples of mutational spectra at time points subsequent to the instauration of a regimen of disturbance are clearly distinguishable from those derived from the undisturbed individual (Figure 1 a&b). With a frequency depending on the intensity of the disturbance, some clonal populations will fail to collapse, and so they will eventually fulfill the diagnostic criteria of tumor (Fig.1c). We find that every simulation that expands an allele above a threshold t_2 went on to reach the tumor state (Supplemental).

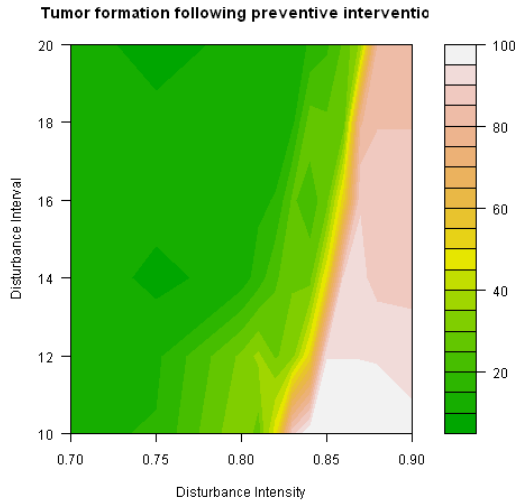


Fig. 2. A cohort of samples is studied in which tumors form in the presence of a disturbance intensity (p_{Δ}) of 0.9 and an interval between disturbances (Δ_k) of 10. During re-runs of the simulations, preventive interventions are applied through the reduction of disturbance once a threshold (selected as the maximum reached in a large cohort of normal undisturbed simulations) is reached in the total mutation load. The effectiveness of tumor prevention by reduction of disturbance intensity (p_{Δ}) and interval between disturbances (Δ_k) is displayed. The relative number of times (%) that a tumor formed are plotted against the reduced intensity (modified p_{Δ}) following intervention on the x axis and the modified interval (Δ_k) between disturbances following the intervention on the y axis. The risk of tumor decreases with reduction in the disturbance. There is a relatively steep drop off with nearly complete ablation of the development of tumors past a boundary which is a function of the disturbance interval and intensity.

Thus repeated monitoring of the mutational spectra for an individual simulation can forecast the emergence of a tumor and thus be used as an early detection tool (Figure 2c). In keeping with the intermediate disturbance hypothesis, the simulations reveal that for a given mutation rate, intermediate disturbance is more effective in causing tumors than either more extreme high or low levels. Moreover this effect can be observed throughout a wide range of mutation rate (Figure 2).

4 Conclusions

The Montebello Model of tumor formation enables wide exploration of the parameter spaces that influence the emergence of tumors from a tissue at risk including the balance between proliferation and death, mutational rate, and disturbance. It provides a tool to test the interplay of evolutionary factors in the context of metapopulation dynamics, and it shows how disturbance can act as a powerful carcinogen. The simulations also demonstrate how biometrics, capturing variational dynamics among cell populations, can be used to stratify simulated populations according to risk level, monitor cancer risk and assess the effectiveness of preventive measures that interfere with disturbance.

References

1. Beerenwinkel, N., et al.: Genetic progression and the waiting time to cancer. *PLoS Comput. Biol.* 3, e225 (2007)
2. González-García, I., Solé, R.V., Costa, J.: Metapopulation dynamics and spatial heterogeneity in cancer. *Proc. Natl. Acad. Sci. USA* 99, 13085–13089 (2002)
3. Tarafa, G., et al.: Mutational load distribution analysis yields metrics reflecting genetic instability during pancreatic carcinogenesis. *Proc. Natl. Acad. Sci. USA* 105, 4306–4311 (2008)
4. Mantovani, A., et al.: Cancer-related inflammation. *Nature* 454, 436–444 (2008)
5. Merlo, L.M.F., et al.: Cancer as an evolutionary and ecological process. *Nat. Rev. Cancer* 6, 924–935 (2006)
6. Vincent, T.L., Gatenby, R.A.: An evolutionary model for initiation, promotion, and progression in carcinogenesis. *Int. J. Oncol.* 32, 729–737 (2008)
7. Buckling, A., et al.: Disturbance and diversity in experimental microcosms. *Nature* 408, 961–964
8. Hanski, I.: *Metapopulation Biology* (1997)
9. Tilman, D., May, R.M., Lehman, C.L., Nowak, M.A.: Habitat destruction and the extinction debt. *Nature* 371, 65–66 (1994)
10. Frank, S.: *Cancer Kinetics* (2007)

Towards Generalized Measures Grasping CA Dynamics

Jan M. Baetens* and Bernard De Baets

KERMIT, Department of Applied Mathematics, Biometrics and Process Control,
Ghent University, Coupure links 653, Gent, Belgium
{jan.baetens,bernard.de.baets}@ugent.be

Abstract. In this paper we conceive Lyapunov exponents, measuring the rate of separation between two initially close configurations, and Jacobians, expressing the sensitivity of a CA's transition function to its inputs, for cellular automata (CA) based upon irregular tessellations of the n -dimensional Euclidean space. Further, we establish a relationship between both that enables us to derive a mean-field approximation of the upper bound of an irregular CA's maximum Lyapunov exponent. The soundness and usability of these measures is illustrated for a family of 2-state irregular totalistic CA.

Keywords: irregular tessellation, Jacobian, Lyapunov exponent.

1 Introduction

Since their conceptualization by von Neumann [31] more than 60 years ago, cellular automata (CA) have proven their usefulness in applied sciences as adequate modeling tools in numerous scientific fields, such as epidemiology [21,33], demography [7,8,15], microbiology [23,24], traffic engineering [9], hydrology and geology [10,12,13,22,29], and numerous others [11,18,20,25], while in exact sciences much attention has been given to the complex spatio-temporal dynamics of these intrinsically simple discrete dynamical systems [6,26,35,36,37]. In contrast to continuous dynamical systems such as ordinary and partial differential equations (ODE and PDE) that often allow to investigate the system's stability properties without having to solve the ODE or PDE, adequate conclusions about a CA's dynamical properties can mostly only be drawn from extensive computer simulations [17,35,37], except for the class of additive CA [34]. This has motivated several researchers to develop quantitative measures for discriminating between the behavioral classes of CA distinguished by Wolfram [35], such as the Hamming distance [6,35], the Langton parameter [19], Lyapunov exponents [11,27,28], entropies and dimensions [16], and others [38], or by relying on mean-field approximations [14].

Among these measures, Lyapunov exponents that were first introduced in 1D CA as the propagation speed of the damage front originating from an initial perturbation of the state of one of the cells [36], and later described rigorously for

* Corresponding author.

1D CA [26], are perhaps the most promising as indicated by their prevalent use in papers on the phenomenology of CA [4,5,6,11,26,27,28]. Besides, their manifold use for the characterization of continuous dynamical systems makes them easily accessible for researchers that are not acquainted with the typicalities of CA. Most frequently, Lyapunov exponents have been applied to characterize the dynamics of 1D CA, since in this case, the damage front can propagate only to the left or to the right of the initially perturbed cell, enabling a sound formulation of right and left Lyapunov exponents [26]. Clearly, if higher-dimensional CA are at stake, the usefulness of such directional Lyapunov exponents is strongly hampered since the damage front in, for instance 2D CA, can propagate circularly from an initial perturbation. For that reason, Bagnoli et al. [6] formulated a non-directional Lyapunov exponent, which has been applied in combination with a measure expressing the sensitivity of a CA's transition function to its inputs and based upon Boolean derivatives [30], to study 1D CA, as well as 2D lattice gas automata [4,5]. However, the latter measure is formulated in such a way that its use is limited to CA based upon regular tessellations of \mathbb{R}^n since it assumes that the neighborhood structure is fixed, which is clearly not the case if irregular tessellations are at stake. Hence, in order to grasp the dynamics of CA based upon irregular tessellations, described rigorously in [2], as well as to compare the dynamics of CA defined upon different tessellations of \mathbb{R}^n , a generalized definition of this measure should be formulated, and its relationship with the non-directional Lyapunov exponents should be readdressed.

In Section 2 we outline the mathematical preliminaries that are necessary for a proper understanding of Section 3 in which we conceive Lyapunov exponents and Jacobians for irregular CA, and establish a measure grasping the CA's sensitivity to its input by relying on the latter. An exemplary simulation study of 2-state irregular totalistic CA concludes this paper.

2 Preliminaries

We state the definition of a cellular automaton on an arbitrary tessellation of a n -dimensional Euclidean space, which constitutes an extension to the classical CA paradigm that predominantly relies on regular tessellations of \mathbb{R}^n ever since von Neumann's pioneering work [32].

Definition 1. (*Cellular automaton*)

A cellular automaton (CA) \mathcal{C} can be represented as a sextuple

$$\mathcal{C} = \langle \mathcal{T}, S, s, s_0, N, \Phi \rangle ,$$

where

- (i) \mathcal{T} is a countably infinite tessellation of a n -dimensional Euclidean space \mathbb{R}^n , consisting of cells c_i , $i \in \mathbb{N}$.
- (ii) S is a finite set of k states, often $S \subset \mathbb{N}$.
- (iii) The output function $s : \mathcal{T} \times \mathbb{N} \rightarrow S$ yields the state value of cell c_i at the t -th discrete time step, i.e. $s(c_i, t)$.

(iv) The function $s_0 : \mathcal{T} \rightarrow S$ assigns to every cell c_i an initial state, i.e. $s(c_i, 0) = s_0(c_i)$.

(v) The neighborhood function $N : \mathcal{T} \rightarrow \bigcup_{p=1}^{\infty} \mathcal{T}^p$ maps every cell c_i to a finite

sequence $N(c_i) = (c_{i_j})_{j=1}^{|N(c_i)|}$, consisting of $|N(c_i)|$ distinct cells c_{i_j} .

(vi) $\Phi = (\phi_i)_{i \in \mathbb{N}}$ is a family of functions

$$\phi_i : S^{|N(c_i)|} \rightarrow S,$$

each ϕ_i governing the dynamics of cell c_i , i.e.

$$s(c_i, t + 1) = \phi_i(\tilde{s}(N(c_i), t)),$$

where $\tilde{s}(N(c_i), t) = (s(c_{i_j}, t))_{j=1}^{|N(c_i)|}$.

Although the complexity measures in the subsequent sections are derived for any CA that obeys the former definition, in the simulation study presented in the final section of this paper we focus on a family of totalistic CA, which we define as follows.

Definition 2. (Totalistic cellular automaton)

A totalistic cellular automaton (CA) is a CA for which $S \subset \mathbb{N}$, and for which there exists a $\Omega : \mathbb{N} \rightarrow S$ such that

$$s(c_i, t + 1) = \phi_i(\tilde{s}(N(c_i), t)) = \Omega(\sigma_i),$$

where $\sigma_i = \sum_{j=1}^{|N(c_i)|} s(c_{i_j}, t)$.

For the sake of uniformity, we also set up an enumeration scheme for irregular totalistic CA in accordance with the enumeration developed for their regular counterparts. Such a concise enumeration scheme allows to identify every ϕ_i that can be formulated, given the number of possible states k , by means of a unique number, commonly referred to as the rule number. To overcome the unboundedness of σ_i that arises from allowing irregular tessellations of \mathbb{R}^n , we introduce an upper bound θ on σ_i such that $\Omega(\sigma_i) = \Omega(\theta)$ if $\sigma_i \geq \theta$. As such, the rule number for a k -state, θ -sum irregular totalistic CA, denoted R_θ^T , can then be found from its base- k representation, containing $\mu = \theta + 1$ digits, $z_\theta z_{\theta-1} \dots z_2 z_1 z_0$ as

$$R_\theta^T = z_\theta k^{\mu-1} + z_{\theta-1} k^{\mu-2} + \dots + z_2 k^{\mu-(\mu-2)} + z_1 k^{\mu-(\mu-1)} + z_0, \quad (1)$$

where $z_f \in \{0, 1, \dots, k-1\}$ represents the state value assigned to c_i at the $t+1$ -th time step if $\sigma_i = f$. A total of $k^{\theta+1}$ different rules can be enumerated for this family of irregular CA.

3 Lyapunov Exponents and Jacobians for Irregular CA

3.1 Lyapunov Exponents

Let s_0 and s_0^* be two initial configurations of a 2-state CA for which $S = \{0, 1\}$, such that there is only one $c_i \in \mathcal{T}$ for which $s_0(c_i) \neq s_0^*(c_i)$, i.e. s_0^* constitutes

the smallest possible perturbation of s_0 . In what follows, we will refer to a cell c_i for which $s(c_i, t) \neq s^*(c_i, t)$ as a defective or perturbed cell, or in short, as a defect. Further, if we define ϵ_t as the number of defective cells at the t -th time step, *i.e.*

$$\epsilon_t = |\{i \mid s(c_i, t) \neq s^*(c_i, t)\}|, \tag{2}$$

the quantity

$$\lambda = \lim_{t \rightarrow \infty} \frac{1}{t} \log \left(\frac{\epsilon_t}{\epsilon_0} \right), \tag{3}$$

can be intuitively interpreted as a maximum Lyapunov exponent (MLE). Yet, as indicated by Bagnoli et al. [6], we must take into account that a CA’s discrete nature can cause defects to annihilate each other such that the quantity given by the right-hand side of Eq. (3) approaches zero as $t \rightarrow \infty$. Indeed, we observed this tendency for the family of irregular CA covered in this paper. As suggested by Bagnoli et al. [6], this artifact can be overcome by keeping track of the evolution of all the defects that arise during the CA’s evolution, though they neglected to include an algorithmic procedure that allows a proper evaluation of ϵ_t , and hence of λ . For that reason, we provide a brief algorithmic procedure that enables the calculation of the non-directional Lyapunov exponent of a CA (Algorithm 1).

Algorithm 1. Procedure for calculating the Lyapunov exponent of an irregular CA

```

Create two initial configurations  $s(\cdot, 0)$  and  $s^*(\cdot, 0)$  such that  $\epsilon_0 = 1$  ;
Calculate  $s(\cdot, 1)$  and  $s^*(\cdot, 1)$ ;
Determine the set  $\mathcal{D}_1 = \{c_j \in \mathcal{T}^* \mid s^*(c_j, 1) \neq s(c_j, 1)\}$ ;
For any  $c_j \in \mathcal{D}_1$ , create a replica  $s^j(\cdot, 1)$  of  $s(\cdot, 1)$ , and perturb it such that
 $s^j(c_j, 1) = s^*(c_j, 1)$ ;
Store these perturbed configurations, denoted as  $s^{*j}(\cdot, 1)$ , in a set  $A$ ;
foreach time step  $t$  do
    Calculate  $s(\cdot, t+1)$  and  $s^{*j}(\cdot, t+1)$ ;
    Construct the multiset  $\mathcal{E}_{t+1} = \{(c_k, m(c_k)) \mid c_k \in \mathcal{D}_{t+1}\}$  where
     $\mathcal{D}_{t+1} = \{c_k \in \mathcal{T}^* \mid \exists j : s^{*j}(c_k, t+1) \neq s(c_k, t+1)\}$  and  $m(c_k)$  gives the
    number of replicas for which  $s^{*j}(c_k, t+1) \neq s(c_k, t+1)$ ;
    Calculate  $\epsilon_{t+1}$ ;
    Delete all the elements in  $A$ ;
    Construct for any  $c_j \in \mathcal{E}_{t+1}$  a replica  $s^j(\cdot, t+1)$  of  $s(\cdot, t+1)$  and perturb it
    in such a way that  $s^j(c_j, t+1) = s^*(c_j, t+1)$ ;
    Store these perturbed configurations, denoted as  $s^{*j}(\cdot, t+1)$ , in the set  $A$ ;
end
Calculate  $\lambda$  using Eq. (3);

```

It should be emphasized that, notwithstanding Eq. (3) demands to evaluate λ as $t \rightarrow \infty$, practical considerations make us to calculate the MLE for finite T , and for finite tessellations \mathcal{T}^* of a compact subset of \mathbb{R}^n . The MLE has been used before to classify both elementary and totalistic 1D cellular automata [6,4],

and is applied in our work to quantitatively describe the dynamics of 2-state, 5-sum irregular totalistic CA.

3.2 Jacobians

In order to express the sensitivity of a CA’s transition function ϕ_i to its input $\tilde{s}(N(c_i), t)$, we can construct a Jacobian matrix J that has $|\mathcal{T}^*| \times |\mathcal{T}^*|$ entries:

$$J_{ij} = \begin{cases} \frac{\partial s(c_i, t + 1)}{\partial s(c_j, t)}, & \text{if } c_j \in N(c_i), \\ 0 & \text{, else,} \end{cases} \tag{4}$$

where $\frac{\partial s(c_i, t + 1)}{\partial s(c_j, t)}$ is the Boolean derivative, introduced in CA by Vichniac [30]. If altering $s(c_j, t)$ affects $s(c_i, t + 1)$, this Boolean derivative equals one, whereas it equals zero if such an alteration has no influence on the outcome of $\phi_i(\tilde{s}(N(c_i), t))$. In contrast with the Jacobian of an elementary CA, the Jacobian of a CA based upon irregular tessellations of \mathbb{R}^n is not tridiagonal.

Considering the variability of $|N(c_i)|$ in irregular CA, the average proportion of cells c_j in $N(c_i)$ that affects $s(c_i, t + 1)$ is given by

$$\mu(t) = \frac{1}{|\mathcal{T}^*|} \sum_{c_i} \frac{1}{|N(c_i)|} \sum_{j=1}^{|N(c_i)|} J_{ii_j}. \tag{5}$$

Essentially, $\mu(t)$ expresses the sensitivity of a CA’s transition function to its inputs. Its geometric mean $\bar{\mu}$ after a large number of time steps T is

$$\bar{\mu} = \left(\prod_{t=1}^T \mu(t) \right)^{\frac{1}{T}}. \tag{6}$$

Understandably, higher values of $\bar{\mu}$ indicate a higher sensitivity of ϕ_i to its input $\tilde{s}(N(c_i), t)$. Since the sensitivity is, within the outer summation of Eq. (5), normalized for every c_i with respect to $|N(c_i)|$, it can be used to characterize a CA regardless of the tessellation it is based upon. Hence, it is an appropriate measure that can be exploited to compare the dynamics of CA that are based upon the same transition function ϕ_i , but employ different tessellations of \mathbb{R}^n .

3.3 Assessing an Upper Bound for Lyapunov Exponents of Irregular CA

If we define the mean connectivity of \mathcal{T}^* as

$$\bar{V} = \frac{1}{|\mathcal{T}^*|} \sum_{c_i} |N(c_i)|, \tag{7}$$

and we indicate that the right-hand side Eq. (3) measures the average rate of separation of two trajectories in phase space during one time step, we may argue

that $\overline{V} \bar{\mu}$ represents a mean-field approximation for the maximum number of cells c_j with $c_j \in N(c_i)$ that are affected by a defect in c_i during one subsequent time step. Accordingly, we can define a function that maps a given $\bar{\mu}$ to the upper bound on the MLE (λ_m), *i.e.*

$$\lambda_m(\bar{\mu}) = \log(\overline{V} \bar{\mu}) \tag{8}$$

yields a mean-field approximation of $\lambda_m(\bar{\mu})$ if both $\bar{\mu}$ and \overline{V} are known. We refer to the outcome of Eq. (8) as a mean-field approximation, since it is derived using the mean connectivity \overline{V} , hence, it makes of the discrepancies between $|N(c_i)|$, and it is only valid for $t \rightarrow \infty$. Clearly, the function given by Eq. (8) reaches its maximum for $\bar{\mu} = 1$, which can occur if and only if all cells c_j in $N(c_i)$ are affected by a defect present in c_i at the t -th time step during one subsequent time step, and this holds for all c_i in \mathcal{T}^* . Furthermore, since $\bar{\mu}$ is confined between zero and one, it is clear that the upper bound for the MLE drops as ϕ_i becomes less sensitive, expressed in terms of $\bar{\mu}$, to its input $\tilde{s}(N(c_i), t)$.

4 Phenomenological Study of Irregular Totalistic CA

4.1 Conventions

Unless stated otherwise, the results presented in this section were obtained numerically for $T = 500$, since by then both λ and $\bar{\mu}$ showed convergence in the sense that an increase of T did not significantly alter the numerically assessed values. Furthermore, periodic boundary conditions were applied in order to minimize boundary effects owing to the finiteness of \mathcal{T}^* that, for the simulations considered in this section, consisted of 675 irregular cells covering a unit square, and were generated from random seeds in $[0, 1]^2$ using a Voronoi tessellation. Using a Moore neighborhood, we obtained for this exemplary tessellation $\overline{V} = 6.97$. As an exemplary family of irregular totalistic CA we consider in the remainder of this section 2-state, 5-sum irregular totalistic CA, for which $S = \{0, 1\}$. Hence, in accordance with the enumeration scheme for irregular totalistic CA outlined in Section 2.64 CA rules can be enumerated within this particular CA family.

4.2 Phenomenology

Figure 1 depicts the numerically evaluated MLE (λ) versus the geometric mean of the proportion of non-zero entries in $J(\bar{\mu})$ of the 2-state, $\theta = 5$ irregular totalistic CA for which $\lambda \neq -\infty$, together with the function $\lambda_m(\bar{\mu}) = \log(\overline{V} \bar{\mu})$, which, according to Eq. (8), for the tessellation used in these simulations equals $\lambda_m(\bar{\mu}) = \log(6.97 \bar{\mu})$. Besides, it displays a logarithmic function $\lambda_m(\bar{\mu}) = \log(\overline{V}^* \bar{\mu})$ that was fitted to $(\bar{\mu}, \lambda)$ data pairs. Since the $(\bar{\mu}, \lambda)$ enclosed within the demarcated area depicted in Fig. 1 clearly deviate from the overall trend that can be inferred from this figure, these data pairs were excluded from the fitting procedure. As such, we established $\overline{V}^* = 6.777$ with a coefficient of determination $R^2 = 0.98$.

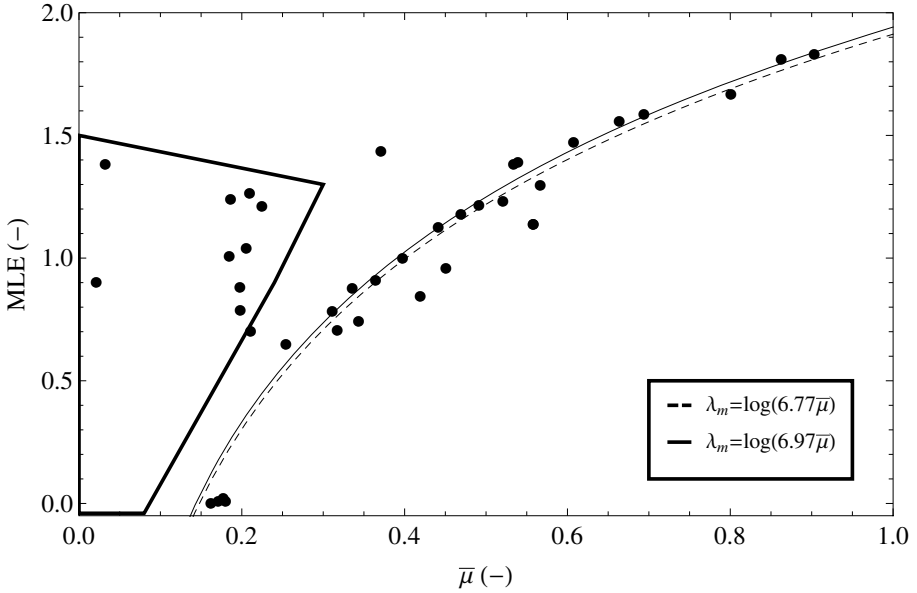


Fig. 1. Maximum Lyapunov exponent (λ) versus the geometric mean of the proportion non-zero entries in $J(\bar{\mu})$ after 500 time steps, starting from a random initial condition. Results are only shown for those 2-state, $\theta = 5$ irregular totalistic CA for which $\lambda \neq -\infty$.

The close agreement between the fitted function and the function given by Eq. (8) indicates the validity of the latter.

Though Figure 1 confirms the validity of Eq. (8), it also shows that the overall upper bound on the MLE, which is given by Eq. (8) for $\bar{\mu} = 1$ such that $\lambda_m(\bar{\mu}) = \log(6.97) \approx 1.94$, is not attained by any of the 64 considered 2-state, 5-sum irregular totalistic CA. More specifically, the highest MLE, equaling 1.83, is found for rule 85, which, at the same time gives rise to the highest $\bar{\mu}$ that is observed among the members of the CA family at stake, being $\bar{\mu} = 0.9$. Hence, given the fact that none of the CA contained in the considered CA family attains the theoretical upper bound on $\bar{\mu}$, *i.e.* $\bar{\mu} = 1$, meaning that there is no CA rule for which $J_{ij} = 1$ for all c_i in \mathcal{T}^* and $c_j \in N(c_i)$, it is obvious that the overall upper bound on the MLE cannot be reached by any of the investigated CA rules. Yet, this finding inevitably gives rise to the question why none of the CA evolves towards $\bar{\mu} = 1$. This issue can be elucidated by reconsidering the upper bound θ that was introduced in Section 2 to set up an enumeration scheme for k -state irregular totalistic CA. This upper bound θ entails all σ_i for which $\sigma_i \geq \theta$ to be mapped to the same state, *i.e.* $\Omega(\sigma_i) = \Omega(\theta)$, and, as such, makes the totalistic CA partially insensitive to its input. For instance, for the CA family at stake, we chose $\theta = 5$, such that $\Omega(\sigma_i) = \Omega(5)$ for all $\sigma_i \geq 5$. Yet, seen the exemplary tessellation has $\bar{V} \approx 7$ the existence of cells $c_i \in \mathcal{T}^*$ for which $\sigma_i \geq 5$ is certainly not unlikely, though this cannot be discerned by the

CA since $\Omega(\sigma_i) = \Omega(5)$. Clearly, the lower the upper bound θ is chosen with respect to \overline{V} , the larger becomes its influence on the CA's dynamical properties.

Table 1, giving an overview of the λ and $\bar{\mu}$ that were assessed numerically for the CA within the studied family of irregular totalistic CA, shows that 26 rules give rise to $\lambda = -\infty$ indicating that these CA evolve converging trajectories in phase space. For comprehensiveness, we must underline that, in the framework of this preliminary study, the values reported in Table 1 are obtained by considering only one perturbed initial configuration s_0^* in order to curtail the amount of computation time, whereas a profound study of a CA's dynamical properties should be based upon an ensemble of perturbed initial configurations [3].

Table 1. Dynamical properties of 2-state, $\theta = 5$ irregular totalistic rules: sensitivity to initial conditions, expressed in terms of λ , and the sensitivity of Ω to its input σ_i , expressed by $\bar{\mu}$

rule	$\bar{\mu}$	λ	rule	$\bar{\mu}$	λ	rule	$\bar{\mu}$	λ	rule	$\bar{\mu}$	λ
0	0	$-\infty$	16	0	$-\infty$	32	0	$-\infty$	48	0.06	$-\infty$
1	0.14	$-\infty$	17	0.44	1.13	33	0.32	0.71	49	0.06	$-\infty$
2	0.56	1.14	18	0.66	1.56	34	0.52	1.23	50	0.02	$-\infty$
3	0.02	$-\infty$	19	0.54	1.39	35	0.34	0.88	51	0.02	$-\infty$
4	0.53	1.38	20	0.86	1.81	36	0.37	1.44	52	0.02	$-\infty$
5	0.21	1.27	21	0.9	1.83	37	0.22	1.22	53	0.02	$-\infty$
6	0.42	0.85	22	0.69	1.59	38	0.21	1.04	54	0.02	$-\infty$
7	0.03	$-\infty$	23	0	$-\infty$	39	0.18	1.01	55	0.02	$-\infty$
8	0.01	$-\infty$	24	0.02	0.9	40	0.19	1.24	56	0	$-\infty$
9	0.61	1.47	25	0.47	1.18	41	0.2	0.89	57	0	$-\infty$
10	0.8	1.67	26	0.57	1.3	42	0.21	0.71	58	0	$-\infty$
11	0.03	1.39	27	0.49	1.22	43	0.2	0.79	59	0	$-\infty$
12	0.4	1.	28	0.31	0.79	44	0.18	0.01	60	0	$-\infty$
13	0.45	0.96	29	0.36	0.91	45	0.18	0.02	61	0	$-\infty$
14	0.34	0.74	30	0.25	0.65	46	0.17	0.01	62	0	$-\infty$
15	0.07	$-\infty$	31	0.08	$-\infty$	47	0.16	0	63	0	$-\infty$

5 Discussion

Despite the validity of Eq. (8) and the usability of Lyapunov exponents and Jacobian-based measures was demonstrated for the family of 2-state, 5-sum totalistic CA, several issues concerning this approach are still awaiting closer inspection. First, the studied CA family encloses not more than 64 rules, whereas the usefulness of the approach should be checked against a much broader family. Second, the conclusions in this paper are drawn from one exemplary irregular tessellation, which makes it debatable whether part of the CA behavior observed is caused by the underlying tessellation, rather than by the CA's intrinsic properties. Third, and closely related to the second issue, concerns the effects on CA

dynamics that may arise from using a regular rather than an irregular tessellation. In forthcoming work we hope to shed some light on each of these issues, and, by doing so, consolidating the approach discussed in this paper.

6 Conclusions

In this paper we proposed adequate measures, namely Lyapunov exponents and Jacobian-based measure, that are able to grasp the dynamics of a CA regardless the tessellation it is based upon, and to provide an objective means for comparing the dynamics of CA across different tessellations of \mathbb{R}^n . Further, the relationship between both allows to obtain a mean-field approximation of the upper bound on a CA's Lyapunov exponent. The soundness of both measures is illustrated by means of a simulation study in which we considered the family of 2-state, 5-sum totalistic CA. In a forthcoming study, we employ both measures to quantitatively describe the dependence of a CA's dynamical properties on exploited tessellation.

Acknowledgments. The authors wish to acknowledge S. Wolfram and his co-workers for their commitment in organizing the yearly New Kind of Science Summer School, which served as a steppingstone for initiating this work.

References

1. Alexandridis, A., Vakalis, D., Siettos, C., Bafas, G.: A cellular automata model for forest fire spread prediction: The case of the wildfire that swept through Spetses Island in 1990. *Appl. Math. Comp.* 204, 191–201 (2008)
2. Baetens, J., De Baets, B.: Cellular automata on irregular tessellations. *Chaos, Solitons Fractals* (2010) (submitted)
3. Baetens, J., De Baets, B.: Phenomenological study of irregular cellular automata based on Lyapunov exponents and Jacobians. *Chaos* (2010) (submitted)
4. Bagnoli, F., Rechtman, R.: Synchronization and maximum Lyapunov exponents of cellular automata. *Phys. Rev. E: Stat. Nonlinear Soft Matter Phys.* 59, R1307–R1310 (1999)
5. Bagnoli, F., Rechtman, R.: Thermodynamic entropy and chaos in a discrete hydrodynamical system. *Phys. Rev. E: Stat. Nonlinear Soft Matter Phys.* 79, 041115 (2009)
6. Bagnoli, F., Rechtman, R., Ruffo, S.: Damage spreading and Lyapunov exponents in cellular automata. *Phys. Lett. A* 172, 34–38 (1992)
7. Baltzer, H., Braun, P., Köhler, W.: Cellular automata models for vegetation dynamics. *Ecol. Modell.* 107, 113–125 (1998)
8. Batty, M.: Cellular dynamics: modelling urban growth as a spatial epidemic. In: Fischer, M., Leung, Y. (eds.) *Geocomputational Modelling*, pp. 109–141. Springer, Berlin (2001)
9. Chopard, B., Luthi, P., Queloz, P.: Cellular automata model of car traffic in a two-dimensional street network. *J. Phys. A: Math. Gen.* 29, 2325–2336 (1996)
10. Coppola, E., Tomassetti, B., Mariotti, L., Verdecchia, M., Visconti, G.: Cellular automata algorithms for drainage network extraction and rainfall assimilation. *Hydrol. Sci. J.* 52, 579–592 (2007)

11. Courbage, M., Kamiński, B.: Space-time directional Lyapunov exponents for cellular automata. *J. Stat. Phys.* 124, 1499–1509 (2006)
12. Crisci, G., Iovine, G., Gregorio, S.D., Lupiano, V.: Lava-flow hazard on the SE flank of Mt. Etna (Southern Italy). *J. Volcanol. Geotherm. Res.* 177, 778–796 (2008)
13. D'Ambrosio, D., Spataro, W.: Parallel evolutionary modelling of geological processes. *Parallel Comput.* 33, 186–212 (2007)
14. Deutsch, A., Dormann, S.: *Cellular Automaton Modeling of Biological Pattern Formation: Characterization, Applications, and Analysis*. Birkhäuser, Bonn (2005)
15. Dewdney, A.: Sharks and fish wage an ecological war on the toroidal planet. *Sci. Am.* 251, 14–22 (1984)
16. Ilachinski, A.: *Cellular Automata. A Discrete Universe*. World Scientific, London (2001)
17. Jackson, E.: *Perspectives of Nonlinear Dynamics*, vol. 1 and 2. Cambridge University Press, Cambridge (1991)
18. Kier, L., Seybold, P., Cheng, C.: *Modelling Chemical Systems using Cellular Automata*. Springer, Dordrecht (2005)
19. Langton, C.: Computation at the edge of chaos. *Physica D* 42, 12–37 (1990)
20. Mallet, D., De Pillis, L.: A cellular automata model of tumor-immune system interactions. *J. Theor. Biol.* 239, 334–350 (2006)
21. Milne, J., Fu, S.: Epidemic modelling using cellular automata. In: *Proc. ACAL 2003*, Canberra, pp. 43–57 (December 2003)
22. Parsons, J., Fonstad, M.: A cellular automata model of surface water flow. *Hydrol. Processes* 21, 2189–2195 (2007)
23. Picioreanu, C., van Loosdrecht, M., Heijnen, J.: Mathematical modeling of biofilm structure with a hybrid differential-discrete cellular automaton approach. *Biotechnol. and Bioeng.* 58, 101–116 (1998)
24. Pizarro, G., Griffeath, D., Noguera, D.: Quantitative cellular automaton model for biofilms. *J. Environ. Eng.* 127, 782–789 (2001)
25. Preziosi, L.: *Cancer Modelling and Simulation*. Chapman & Hall, Boca Raton (2003)
26. Shereshevsky, M.: Lyapunov exponents for one-dimensional cellular automata. *J. Nonlinear Sci.* 2, 1–8 (1991)
27. Tisseur, P.: Cellular automata and Lyapunov exponents. *Nonlinearity* 13, 1547–1560 (2000)
28. Urías, J., Rechtman, R., Enciso, A.: Sensitive dependence on initial conditions for cellular automata. *Chaos* 7, 688–693 (1997)
29. Valette, G., Prévost, S., Lucas, L., Léonard, J.: SoDA project: A simulation of soil surface degradation by rainfall. *Computer Graphics* 30, 494–506 (2006)
30. Vichniac, G.: Boolean derivatives on cellular automata. *Physica D* 45, 63–74 (1990)
31. von Neumann, J.: The general and logical theory of automata. In: Jeffress, L. (ed.) *Cerebral Mechanisms in Behaviour: The Hixon Symposium*, pp. 1–32. Wiley, New York (1951)
32. von Neumann, J., Burks, A.: *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign (1966)
33. White, S., del Rey, A., Sanchez, G.: Modeling epidemics using cellular automata. *Appl. Math. Comput.* 186, 193–202 (2007)
34. Wolfram, S.: Algebraic properties of cellular automata. *Commun. Math. Phys.* 93, 219–258 (1984)
35. Wolfram, S.: Universality and complexity in cellular automata. *Physica D* 10, 1–35 (1984)

36. Wolfram, S.: Cellular Automata and Complexity. Westview Press, Boulder (1994)
37. Wolfram, S.: A New Kind of Science. Wolfram Media Inc., Champaign (2002)
38. Wuensche, A., Lesser, M.: The Global Dynamics of Cellular Automata, vol. 1. Addison-Wesley, London (1992)

Synchronization and Control of Cellular Automata

Franco Bagnoli¹, Samira El Yacoubi², and Raúl Rechtman³

¹ Dip. Energetica and CSDC, Università di Firenze, Via S. Marta 3, Firenze, Italy
also INFN, sez. Firenze

`franco.bagnoli@unifi.it`

² Laboratory of Mathematics, Physics and Systems (LAMPS),
University of Perpignan, 52, Paul Alduy Avenue, 66860 - Perpignan Cedex. France

`yacoubi@univ-perp.fr`

³ Centro de Investigación en Energía,
Universidad Nacional Autónoma de México, 65280 Temixco, Mor., Mexico

`rrs@cie.unam.mx`

Abstract. We study the problem of targeted synchronization of *stable chaotic* extended systems, *i.e.*, systems which are not chaotic in the usual sense, but are unpredictable for finite perturbations. Examples are cellular automata, which are completely discrete dynamical systems. We show that the usual approach may lead to counter intuitive results, but that it is possible to exploit the characteristics of the system in order to reduce the distance between two replicas with less control.

1 Introduction

Control theory is a set of techniques for making a dynamical system behave in a desired way by exerting an external effort. In the case of a minimum effort one speaks of *optimal control*. It is obviously hard to reach the optimum limit, but many investigations are devoted to minimize the control for a desired behavior. In general, the problem of control of a dynamical system may be split into two parts: the driving a system to a target area in phase space, and the stabilization of a trajectory originating from this area.

Chaotic systems are ideal targets for control: their sensitivity to small changes may be exploited to drive them to the target area [1], after which chaos may be suppressed in order to make them follow, for instance, a desired periodic orbit [2].

The problem of driving a chaotic system may be seen as the problem of *synchronizing* a replica with a “drive” system that happen to pass through the desired area. This type of synchronization, that can be called master-slave, identical or replica synchronization [3], is quite different from the “spatial” synchronization [4] investigated in extended systems.

While in the usual studies about synchronization one exerts little attention to the optimization of coupling, when formulated as a control problem this becomes a crucial issue.

Most of the literature about control theory deals with low-dimensional, smooth systems. In this paper we want to introduce the problem of driving (synchronizing) *extended, highly non-linear* dynamical systems. There is a class of systems, termed *stable chaotic* [5] which are not chaotic in the usual sense of the sensitive dependence with respect to infinitesimal perturbations, but are nonetheless unpredictable for finite perturbations. In particular, we shall concentrate on cellular automata (CA), which are discrete, deterministic dynamical systems.

CA are widely used to model many systems in various fields, from computer science to earth sciences, biology, physics, sociology, etc. They are usually defined on a graph or a regular lattice, but may easily be extended to include mobile agents. The modeling of a system using cellular automata is conceptually much simpler than those using partial derivatives, and the evolution of such a system is easily performed by a digital computer, without rounding errors. However, for such systems continuity and smoothness (differentiability) do not apply. It is therefore hard to extend the usual techniques used in control theory [7] and to define quantities like Lyapunov exponents and chaotic trajectories. It is however still possible to define the derivatives of discrete systems [8], which prove useful in synchronization investigations [9].

In the case of replica synchronization, the “minimal strength” needed to synchronize a system is related to its chaoticity, defined by the largest Lyapunov exponent in low-dimensional systems. For extended systems, the correspondence between the minimal strength and Lyapunov exponents may break down [6].

In synchronization experiments, the “force” is generally applied blindly, without any relation with the dynamics. The corresponding synchronization effect is analogous to a directed percolation phase transition. The two systems synchronize when their difference goes to zero. Their difference grows due to their “chaotic” dynamics, along the directions identified by the Jacobian matrix of the evolution rule. The synchronization “pressure” reduces the paths along which a difference can propagate. When this reduction overcomes the chaotic growth, the system synchronizes.

In control problems, one wants to exploit the knowledge about a system. It is therefore analogous to a synchronization problem of two different systems with a “targeted” force, that tries to “kill” the growing directions of the difference as soon as possible. We show how the concept of Boolean derivative and that of Boolean Jacobian matrix can be used to achieve this goal.

2 Definitions

Let us start our presentation by considering two smooth, chaotic maps

$$x' = f(x), \quad y' = g(y) + p(f(x) - g(y)),$$

where p is the control “strength”, f and g are two maps, x is valued at the discrete time t , and x' is valued at $t + 1$ (the same applies to y and y'). The x map is the “master” and the y map is the “slave”. The separation between both

maps is $u = x - y$ and the goal of control is to keep $h = |u|$ below a certain threshold (which may be zero), using the minimum strength p . The two maps f and g may be different (for instance, they may use different parameters) or the same, in this case the synchronized state $x = y$ is absorbing.

If the desired trajectory is a natural one for the slave, control is equivalent to replica synchronization.

$$x' = f(x), \quad y' = f(y) + p(f(x) - f(y)), \quad u' = (1 - p)(f(x) - f(y)),$$

where $u = x - y$. For smooth maps, near the synchronization threshold p_c , it is possible to expand $y(t)$ around the unperturbed trajectory $x(t)$,

$$u' = (1 - p)(f(x) - f(y)) \simeq (1 - p) \frac{df(x)}{dx} u.$$

By iterating this map, one obtains the relation between synchronization threshold p_c and Lyapunov exponent λ , $p_c = 1 - \exp(-\lambda)$. The synchronized state is absorbing, since if for some time $x(t) = y(t)$, then the control can be relaxed and the trajectories stay synchronized. However, for chaotic systems, this state is unstable.

Natural systems, however, are rarely low-dimensional. We can extend the previous analysis by considering a lattice of coupled maps, that may be thought as a stroboscopic view of a continuous system:

$$x_i(t + 1) = f(g(x_{i-1}(t), x_i(t), x_{i+1}(t))). \quad (1)$$

where $i = 1, \dots, N$. The function g represents the coupling in the “space”, it can be diffusive (linear) or highly nonlinear. The function f is the individual map, and can lead, when uncoupled, either to fixed points, stable cycles or chaotic oscillations. A perturbation may amplify exponentially in time by the action of f , but only linearly in time through the coupling (propagation to neighboring sites).

The dynamical properties of an extended system are generally analyzed by means of the Lyapunov spectrum. Using vector notation, Eq. (1) can be written as

$$\mathbf{x}(t + 1) = \mathbf{F}(\mathbf{x}(t)), \text{ and } J_{ij}(\mathbf{x}(t)) = \frac{\partial F_i(\mathbf{x}(t))}{\partial x_j}$$

are components of the Jacobian matrix of \mathbf{F} , $i, j = 1, \dots, N$. For an infinitesimal perturbation

$$\delta(t + 1) = \mathbf{J}(\mathbf{x}(t))\delta(t).$$

For instance, the Jacobian matrix of one-dimensional systems with nearest neighbor couplings has zero values except on the three central diagonals.

The eigenvectors of the Jacobian define the instantaneous *tangent space* of a dynamical system. The eigenvalues a_i of the time product $\prod_{t=0}^T \mathbf{J}(\mathbf{x}(t))$ of the Jacobian matrices over a trajectory define the Lyapunov spectrum $\lambda_0 \geq \lambda_1 \geq \lambda_2 \dots$ with $\lambda_i = \log(a_i)/T$ [10]. It is generally assumed that a system is chaotic if $\lambda_0 > 0$ and stable if $\lambda_0 < 0$.

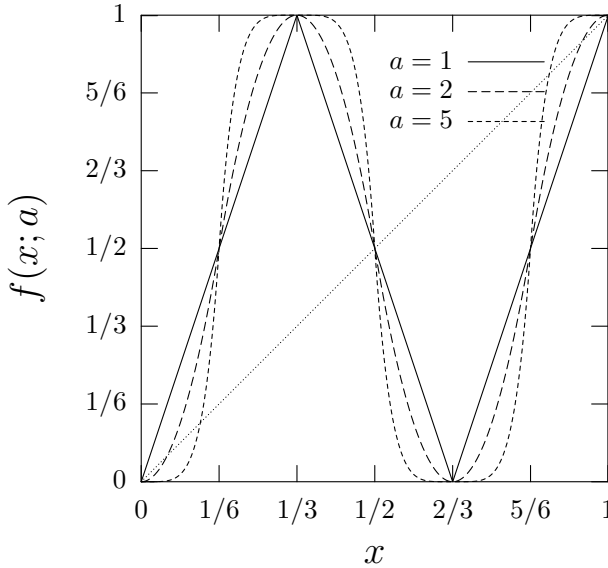


Fig. 1. The map f_{10} for different values of the parameter a . For $a > 1$, this map exhibits two attracting superstable fixed points.

The largest Lyapunov exponent λ_0 (LLE) does not capture all the chaotic characteristics of an extended system. In general, a weak diffusive coupling reduces the LLE (since diffusion limits the exponential expansion along tangent space). Therefore, for small couplings, the maximum of chaoticity corresponds to *uncoupled* maps, but this situation may correspond to the easiest synchronizability (see Section 3).

The scenario of extended systems may be more complex as we discuss below. Consider the map f_{10} shown in Figure 1. This map is obviously stable for $a > 1$, with two fixed points $x_0 = 0$ and $x_1 = 1$, with interleaved basins that act as a sort of “frustration” when coupled,

$$x_i(t + 1) = f \left(\frac{x_{i-1}(t) + x_i(t) + x_{i+1}(t)}{3} \right), \tag{2}$$

$i = 1, \dots, N$ with periodic boundary conditions. The system continues to be stable, exhibiting transient chaos (see Figure 2). After a transient, the eigenvalues of the Jacobian matrix go to zero, and all Lyapunov exponents go to $-\infty$. The long-time evolution of the system is that of the cellular automaton rule 150 (see Section 3), and is unpredictable for *finite* perturbations greater than $1/6$ [11]. A similar behavior can be found in other continuous systems without direct correspondence to cellular automata [5]. Unpredictable stable systems are interesting since the synchronized state is stable.

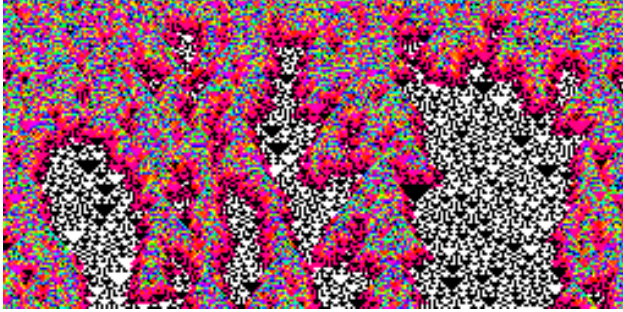


Fig. 2. Time evolution (downward) of a lattice of coupled maps for $a = 2$. Color code: white=0, black=1, gray/color=intermediate values.

3 Cellular Automata and Synchronization

Cellular automata (CA) are completely discrete systems, defined as in Eq. (II), where x_i and f can assume values in a discrete set. In particular, we shall limit our study to Boolean CA, for which the set of discrete values is $\{0, 1\}$. Since the function f is discrete, it can be defined by means of a complete enumeration of outputs given all possible inputs (look-up table). We shall denote by r the size of the neighborhood, *i.e.*, the number of cells whose state constitutes an input for the function f . Eq. (II) corresponds to $r = 3$. The case for which the function f is symmetric with respect to all inputs defines *totalistic* CA, since in this case one can consider that the value of the function f depends only on the sum of the values of sites in the neighborhood. While generic CA with range r are defined by 2^r entries in the look-up table, totalistic CA are defined by $r + 1$ entries. By arranging the output values of the look-up table as Boolean digits, one can compactly represent a CA rule as an integer number R .

Cellular automata may exhibit a large variety of dynamical behaviors. The number of possible states of a lattice of L Boolean cells is finite, and equal to 2^L . Since the dynamics is deterministic, only limit cycles attractors are possible. One can divide the possible scenarios according with the number of attractors, the distribution of their basins and their period. For instance with $r = 3$, trivial rules like rule 0 have only one attractor with a large basin and period equal to one. The identity rule (which is not totalistic) has a large number of attractors (2^L), each one with one state and period 1. The majority rule $1100|_2 = 12|_{10}$ has an intermediate number of fixed-point attractors with short transients (1). “Chaotic” rules like rule $1010|_2 = 10|_{10}$ exhibit cycles with very long period of the order of the total number of possible configurations as in Figure 3. Since in this case the period scales as an exponential of the size of the system, the difference between a periodic and aperiodic trajectory is not relevant for large systems (statistical quantities take similar values). Moreover, a defect or damage typically spreads in the space-time pattern.

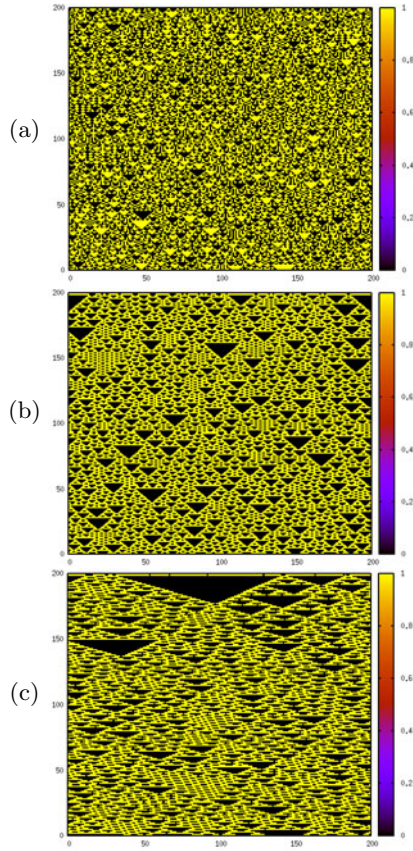


Fig. 3. Typical space-time patterns of “chaotic” rules (time grows downward); (a): $r = 3, R = 10$; (b): $r = 3, R = 6$; (c): $r = 6, R = 30$

It is possible [8] to extend the concept of derivative to cellular automata. The Boolean derivative of \mathbf{F} is the Jacobian matrix with components

$$\begin{aligned}
 J_{i,j} &= \frac{\partial F_i(\mathbf{x})}{\partial x_j} = F_i(x_0, \dots, x_j \oplus 1, \dots, x_{N-1}) \oplus F_i(x_0, \dots, x_j, \dots, x_{N-1}) \\
 &= \begin{cases} 1 & F_i \text{ changes when } x_j \text{ changes,} \\ 0 & F_i \text{ does not change when } x_j \text{ changes,} \end{cases} \quad (3)
 \end{aligned}$$

where \oplus denotes the sum modulo two.

Many “standard” results may be extended to Boolean derivatives, for instance the Taylor expansion

$$f(x, y) = \left(\frac{\partial f}{\partial x} \right)_{x=y=0} x \oplus \left(\frac{\partial f}{\partial y} \right)_{x=y=0} y \oplus \left(\frac{\partial^2 f}{\partial x \partial y} \right)_{x=y=0} xy.$$

One can apply the “linear development” to discrete damages, and define a discrete Jacobian matrix, Eq. (3). Similarly to continuous systems, it is possible to define the largest Lyapunov exponent [9] related to the synchronization threshold (see Section 3). In contrast to continuous dynamics, defects can self-annihilate, so that the actual development of damage is different from the linearized one and they coincide only in the limit of vanishing damage.

There are many ways of “pushing” together two extended replicas. One possibility is “uniform” pushing

$$y_i(t+1) = F_i(\mathbf{y}(t)) + p(F_i(\mathbf{x}(t)) - F_i(\mathbf{y}(t))),$$

for which the analysis presented above applies, with $p_c = 1 - \exp(-\lambda_0)$. This control is however quite difficult to be implemented experimentally in an extended system. Uniform synchronization of chaotic maps gives results similar to low-dimensional systems: $p_c = 1 - \exp(-\lambda_0)$.

Another possibility is that of “pinching” synchronization

$$y_i(t+1) = \begin{cases} F_i(\mathbf{y}(t)) & \text{with probability } 1-p, \\ F_i(\mathbf{x}(t)) & \text{with probability } p. \end{cases}$$

In pinching synchronization, one has the possibility of applying the synchronization “strength” to a suitably chosen subset of sites. Pinching synchronization depends on coupling: uncoupled chaotic maps synchronizes for $p_c = 0$. In general p_c is larger for larger couplings [12].

In synchronization problems, synchronization is applied “blindly”. In control problems, the goal is that of exploiting available information in order to apply a smaller amount of control (or achieve a stronger synchronization).

4 Control of Cellular Automata

We study here the application of synchronization to extended systems

$$\mathbf{x}' = \mathbf{F}(\mathbf{x}), \quad \mathbf{y}' = \mathbf{F}(\mathbf{y}) \oplus \mathbf{p} \odot (\mathbf{F}(\mathbf{x}) \oplus \mathbf{F}(\mathbf{y})),$$

where \odot is the Hadamard (component by component) product, and the effect of synchronization $p_i \in \{0, 1\}$ may depend on the position i . Therefore, the difference \mathbf{u} evolves as

$$\mathbf{u}' = (\mathbf{1} - \mathbf{p}) \odot (\mathbf{F}(\mathbf{x}) \oplus \mathbf{F}(\mathbf{y})) \simeq (\mathbf{1} - \mathbf{p}) \odot \mathbf{J}\mathbf{u}, \quad (4)$$

in the limit of vanishing distance, where the matrix product $\mathbf{J}\mathbf{u}$ is computed modulo two. The control parameter is the average synchronization effort $k = (\sum_i p_i)/N$. The efficacy of synchronization (order parameter) is the asymptotic distance $h = (\sum_i u_i)/N$.

It is possible in principle to find the absolute minimum of k by computing the effects of all possible choices of p_i , given an initial configuration $\mathbf{x}_0 = \mathbf{x}(0)$.

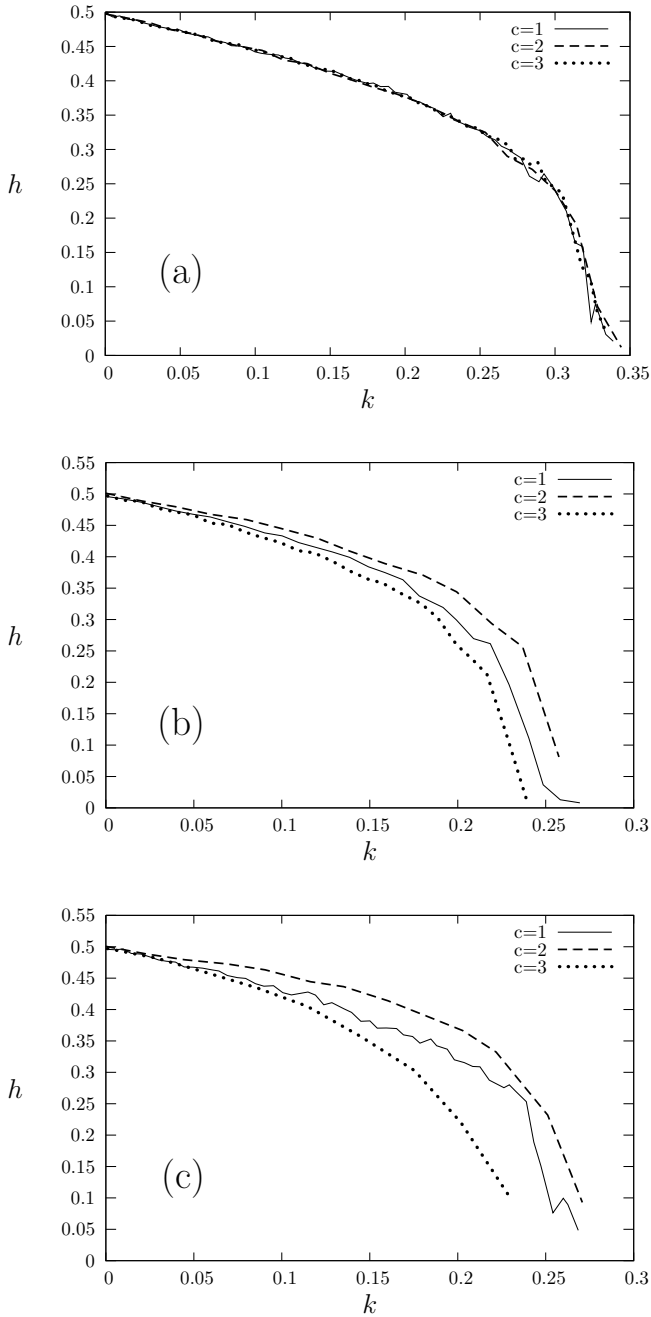


Fig. 4. Plots of the different types of control for: (a) $r = 3, R = 10$ (linear rule); (b) $r = 3, R = 6$ (nonlinear rule); (c) $r = 6, R = 30$ (nonlinear rule). For nonlinear rules, control 2 is worse and control 3 is better than blind one (control type 1).

This constitutes a great computational load. Since we are interested in possible real-time applications, we impose that the choice of $p_i = 1$ may only depend on *local* information: the neighborhood configuration and a $t = 1$ time window.

We investigate three possible way of implementing a control \mathbf{p} : (1) blindly with probability $p = k$ (standard pinching synchronization); (2) with a probability p proportional to the sum of the first-order derivatives and (3) with a probability p inversely proportional to the sum of first-order derivatives.

In order to keep the implementation simple, instead of fixing k and computing the probability p , we let p be a free parameter, and measure the actual fraction of synchronized sites k and the average asymptotic distance h . The previous schemes only require information about \mathbf{x} . If information about \mathbf{y} or about the damage distribution \mathbf{u} is available, the cost k is reduced by a factor h , since in this case we can apply the rule only when it is needed.

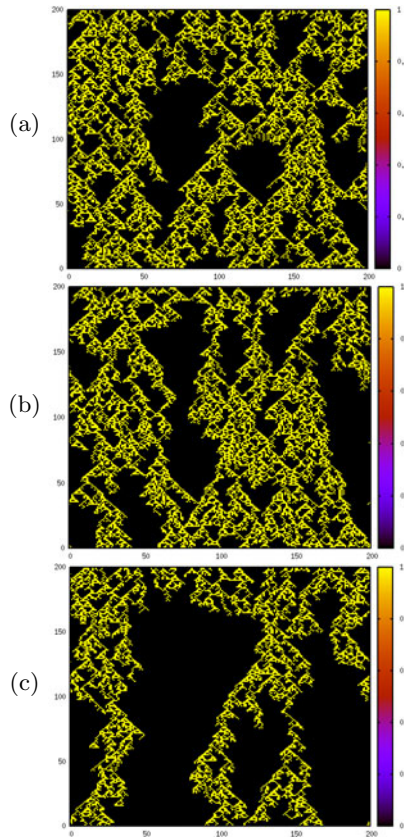


Fig. 5. Time evolution of defects for different types of control; (a): control 1; (b) control 2; (c): control 3, for $r = 3$ and $R = 6$, starting from the same configuration. The effective probability p has been chosen so to have the same average control k in the three cases. Clusters of defects for control 3 (c) are less dense than that of control 1 (a) and 2 (b).

Simulation results are presented in Figure 4. As expected, for linear rules there is no influence of the type of control, since all configurations have the same number of derivatives. For nonlinear rules, the observed behavior is the opposite of what is expected for continuous systems. Control 2, that minimizes the distance h for vanishing number of damages according to Eq. (4), gives worse results than the blind control 1. Control 3, inversely proportional to the sum of first-order derivatives, gives better results than the blind control 1. This result holds also for larger neighborhoods (Figure 4-c), but not for all rules.

This surprising effect may be due to the fact that defects self-annihilate, as shown in Figure 5. In other words, we can exploit the characteristics of cellular automata (and other stable chaotic systems) in order to achieve a better control by exploiting the local contraction of the evolution rule.

5 Conclusions

Spatially extended stable systems (namely cellular automata) may exhibit unpredictable behavior (finite-distance chaoticity). The pinching synchronization threshold is related to this chaoticity. On the other hand, Boolean derivatives and discrete Lyapunov exponents may be used to characterize this kind of chaos. Synchronization may also be exploited for control in experimental situations. In the control problem one aims at discovering a protocol that keeps the distance h below a certain threshold with the minimum “effort”, given some constraints. We have chosen to investigate the behavior of two control schemes based on the local number of non-zero first-order derivatives, taking as reference the “blind” pinching synchronization protocol.

We have shown that, differently from usual chaotic systems, one can exploit *self-annihilation* of defects to obtain synchronization with a weaker control, corresponding to the case in which the control is inversely proportional to the number of non-zero derivatives.

Acknowledgments

Partial economic support from CONACyT project 25116 is acknowledged.

References

1. Shinbrot, T., Ott, E., Grebogy, C., Yorke, J.A.: Phys. Rev. Lett. 65, 3215 (1990)
2. Ott, E., Grebogy, C., Yorke, J.A.: Phys. Rev. Lett. 64, 1196 (1990)
3. Pecora, L.M., Carroll, T.L.: Phys. Rev. Lett. 64, 821 (1990)
4. Rosenblum, M.G., Pikovsky, A.S., Kurths, J.: Phys. Rev. Lett. 76, 1804 (1996)
5. Politi, A., Livi, R., Oppo, G.-L., Kapral, R.: Europhys. Lett. 22, 571 (1993)
6. Bagnoli, F., Baroni, L., Palmerini, P.: Phys. Rev. E 59, 409 (1999)
7. El Yacoubi, S.: Int. J. System Science 39, 529 (2008)
8. Bagnoli, F.: Int. J. Mod. Phys. C 3, 307 (1992)
9. Bagnoli, F., Rechtman, R.: Phys. Rev. E 59, R1307 (1999)
10. Ott, E.: Chaos in dynamical systems. Cambridge University Press, Cambridge (2002)
11. Bagnoli, F., Rechtman, R.: Phys. Rev. E 73, 026202 (2006)
12. Bagnoli, F., Cecconi, F.: Phys. Lett. A 260, 9 (2001)

Discovery by Genetic Algorithm of Cellular Automata Rules for Pattern Reconstruction Task

Anna Piwonska^{1,*} and Franciszek Seredynski^{2,3}

¹ Bialystok University of Technology,
Computer Science Faculty,
Wiejska 45A, 15-351 Bialystok, Poland
a.piwonska@pb.edu.pl

² Institute of Computer Science,
Polish Academy of Sciences,
Ordona 21, 01-237 Warsaw, Poland

³ Polish-Japanese Institute of Information Technology,
Koszykowa 86, 02-008 Warsaw, Poland
sered@ipipan.waw.pl

Abstract. This paper presents results of the study on application of two-dimensional, three-state cellular automata with von Neumann neighborhood to perform pattern reconstruction task. Searching efficient cellular automata rules is conducted with use of a genetic algorithm. Experiments show a very good performance of discovered rules in solving the reconstruction task despite minimum radius of neighborhood and only partial knowledge about neighborhood states available. The paper also presents interesting reusability possibilities of discovered rules in reconstructing patterns different but similar to ones used during artificial evolution.

Keywords: cellular automata, pattern reconstruction task, genetic algorithm.

1 Introduction

Cellular automata (CAs) [11] are discrete dynamical systems studied in many science disciplines, including computability theory, mathematics, physics, theoretical biology, etc. CA consists of identical cells arranged in a regular grid, in one or more dimensions. Each cell can take one of a finite number of states and has an identical arrangement of local connections with other cells called a neighborhood, which also includes the cell itself. After determining initial states of all cells (an initial configuration of a CA), states of cells are updated synchronously according to a local rule defined on a neighborhood. When a grid size is finite, which must be assumed in computer simulations, one must define boundary conditions. There are many possible generalizations of the CAs concept including

* This research was supported by S/WI/2/2008.

other types of rules (e.g. totalistic, probabilistic), other than a rectangular grid (e.g. hexagonal), neighborhood changing in time and many others.

One of the most interesting feature of CAs is that in spite of their simple construction and principle of operation, cells acting together can behave in an inextricable and an unpredictable way. Although cells have a limited knowledge about the system (only its neighbors' states), localized information is propagated at each time step, enabling more global behavior.

The main bottleneck of CAs is a difficulty of constructing CAs rules producing a desired behavior. In some applications of CAs one can design an appropriate rule by hand, based on partial differential equations describing a given phenomenon. However, it is not always possible. In the 90-ties of the last century Mitchell and colleagues proposed to use genetic algorithms (GAs) to discover CAs rules able to perform one-dimensional density classification task [7] and the synchronization task [4]. The results produced by Mitchell et al. were interesting and started development of a concept of automating rule generation using artificial evolution. Breukelaar and Back applied GAs [3] to solve the density classification problem as well as AND and XOR problem in two dimensional CAs. Swiecicka et al. used [10] GAs to find CA rules able to solve multiprocessor scheduling problem. Bandini et al. proposed [1] to use several Machine Learning techniques such as GAs, Support Vector Machines and neural networks to find automatically CA rules able to generate patterns, which are similar in some generic sense to those generated by a given target rule.

In literature one can find several examples of CAs applications in image processing [6,8] as well as evolving by a genetic algorithm CAs rules in image processing task [9]. Some of them deal with image enhancement, detection of edges, noise reduction, image compression, etc.

In this paper we present preliminary results of experiments concerning evolving CAs rules to perform pattern reconstruction task which is related to image processing. The rest of the paper is organized as follows. Section 2 describes pattern reconstruction task in context of CAs. Section 3 presents details of the GA. Experimental results are reported in Section 4. The last section contains conclusions and some remarks about future work.

2 Pattern Reconstruction Task

Digital image reconstruction is a process of restoring true image from its observed but distorted version. Such distortions may include noise, blurred shapes or damage in some regions. Image reconstruction is a very broad field and there are many methods and algorithms concerning this subject described in the literature [2]. This paper focus on one aspect of image reconstruction: complementing missing parts of an image. We will call this task pattern reconstruction since images used in experiments characterize some regularity, easy to observe when looking at them.

We assume that a given pattern is defined on a two-dimensional array (grid of cells) of size 10×10 . Each element of an array can take one of two possible

values: 1 or 2. Let us assume that some fraction q of values of grid elements is not known. These are missing parts of a pattern. Based on such a not complete pattern, it could be difficult to predict unknown states unless one is able to see some dependencies between values of cells. Let us further assume that we have a series of such not complete patterns, created from one given pattern in a random way.

Pattern reconstruction task is formulated as follows. We want to find a CA rule which is able to transform an initial, not complete configuration to the final complete configuration.

Let us construct a two-dimensional CA of size 10×10 , in order to describe our pattern. Our CA will be a three-state: unknown values of grid elements will be represented by state 0. That means that at each time step every cell of our CA can take a value from the set $\{0, 1, 2\}$.

In the context of CAs our task can be described as follows. Let us assume that we have a finite number of random initial configurations, each of which is an incomplete pattern. We want to find a CA rule that is able to converge to a final configuration identical with a complete pattern. That means, a rule that will be able to reconstruct a pattern. We also assume that a complete pattern is not known during searching process. The only data available during searching process is a series of incomplete patterns, randomly created from one given pattern. It is worth mentioning that related to our task is a problem from data mining field described in [5], where a heuristic CA rule was proposed.

Figure 1 presents the example of a pattern of size 10×10 (on the left). Grid elements with value 1 are represented by grey cells and elements with value 2 are represented by black cells. On the right side of this figure one can see the example of this pattern with 50 states unknown. These are represented by white cells. Indexes of unknown elements were generated randomly. Such an incomplete pattern is interpreted as an initial configuration of a CA.

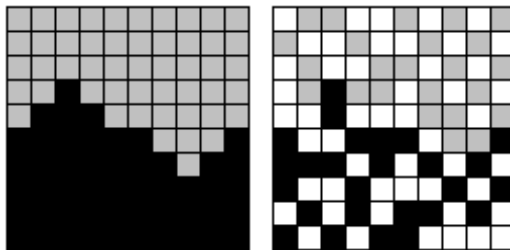


Fig. 1. The examples of a complete pattern (on the left) and an incomplete one (on the right). The incomplete pattern has 50 states unknown ($q = 0.5$).

To search for a CA rule capable of performing pattern reconstruction task, we must first define a neighborhood and boundary conditions. Assuming von Neumann neighborhood, with three possible cell states we have $3^5 = 243$ possible neighborhood states. Thus, the number of possible rules equals to 3^{243} ,

which means enormous search space. In our experiments we assume null boundary conditions: our grid is surrounded by dummy cells always in state 0. The interpretation of this assumption is that we do not know the state of these cells. In fact, they are not a part of our pattern.

3 The GA for Discovering CAs Rules

The proposed CA-based algorithm will run in two phases: the learning phase and the normal operating phase.

3.1 Learning Phase

The purpose of this phase is searching for efficient CAs rules with the use of the GA. The GA starts with a population of P randomly generated 243-bit CA rules. Five cells of von Neumann neighborhood are usually described by directions on the compass: North (N), West (W), Central (C), East (E), South (S). Using this convention, the bit at position 0 in the rule (the top bit in the bar in Fig. 2) denotes a state of the central cell of the neighborhood 0000000 in the next time step, the bit at position 1 in the rule denotes a state of the central cell of the neighborhood 0000001 in the next time step and so on, in lexicographic order of neighborhood.

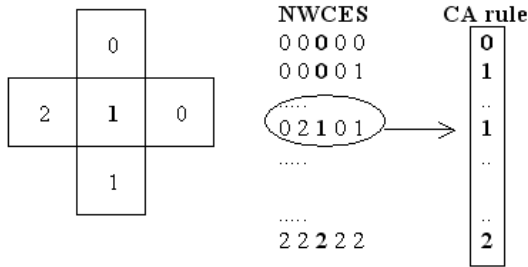


Fig. 2. The neighborhood coding (on the left) and the fragment of the rule - the chromosome of the GA (on the right, in a bar)

The next step is to evaluate individuals in the initial population for the ability to perform pattern reconstruction task. For this purpose, at each generation, starting from a complete pattern, we randomly generate an incomplete pattern with q states unknown. This process proceeds as follows. We have a complete pattern. In single step we randomly select a single cell which has not been previously chosen and this cell changes its state to 0 (unknown state). We repeat these steps until we chose $q \cdot 100$ cells. All these cells will be in the state 0 (unknown).

Then each rule in the population is evolved on that randomly generated incomplete pattern, considered as an initial configuration of CA, for t time steps.

At the final time step we compute the number of cells in the grid, with a state different from 0, that have the correct state. If a given cell is in the state 1 in an initial configuration, then the correct state for this cell in the final configuration is 1. Similarly, if a given cell is in the state 2 in an initial configuration, then the correct state for this cell in the final configuration is 2. The number of cells in a final configuration with the state 1 which are in the correct state will be denoted as n_{-1} and the number of cells in a final configuration with the state 2 which are in the correct state will be denoted as n_{-2} . Since we compute the number of correct states, we deal with maximization problem. The fitness f of a rule i , denoted as f_i , is computed according to formula:

$$f_i = n_{-1} + n_{-2} - n_{-0}, \quad (1)$$

where n_{-0} denotes the number of cells in the final configuration with the state 0. Subtracting the number of cells with the state 0 is a kind of penalty factor and its task is to prevent from evolving to the final configuration with many cells in the state 0. It would be unfavorable situation from the point of view of pattern reconstruction task. The maximal fitness value equals to the number of cells of known states and in the case of a CA composed of 100 cells equals to $100 - q \cdot 100$.

Once we have the genetic representation and the fitness function defined, the GA starts to improve the initial population through repetitive application of selection, crossover and mutation. In our experiments we used tournament selection: individuals for the next generation are chosen through P tournaments. The size of the tournament group is denoted as t_{size} .

After selection individuals are randomly coupled and each pair is subjected to one-point crossover with the probability p_c . If crossover is performed, offspring replace their parents. On the other hand, parental rules remain unchanged.

The last step is a mutation operator. It can take place for each individual in the population with the probability p_m . When a given gene is to be mutated, we replace the current value of this gene by the value 1 or 2, with equal probability. Omitting the value 0 has the same purpose as described previously: it prevents from evolving rules with many 0s. Such rules are more likely to produce configurations containing cells with the state 0. It would be unfavorable situation.

These steps are repeated G generations. The pseudocode of the GA is presented as Algorithm 1.

Algorithm 1: The GA discovering CAs rules

```

input a pattern
create an initial population of P randomly generated rules
for j:=1 to G do
begin
  randomly generate a pattern with q states unknown
  for i:=1 to P do

```

```

begin
  run rule i on this pattern for t time steps
  compute fitness value of rule i
end
for i:=1 to P do
begin
  randomly choose tournament group from the old population
  copy the best rule to the next population
  return chosen rules to the old population
end
randomly couple P individuals
cross each couple with a given probability
mutate rules with a given probability
end

```

3.2 Normal Operating Phase

At the end of the learning phase we have a population of discovered rules which were trained to perform pattern reconstruction task. Let us remind that a complete pattern was not presented during this phase. In each generation, rules learned on an incomplete version of a given pattern. To investigate a real quality of discovered rules, we run each rule on 1000 random initial configurations with q states unknown. For a given final configuration produced by rule i , we count the fraction of cells' states identical with these in a complete pattern. This value, denoted as t_i , is computed according to the formula:

$$t_i = \frac{n_1 + n_2}{100} . \quad (2)$$

An ideal rule will evolve an initial configuration to the final configuration with all cells in correct states. Thus, the maximum value t_i that such an ideal rule can obtain is 1.0. That means that the final configuration is identical with a complete pattern. Since we test each rule on 1000 random initial configurations, the final value for a rule is the rule's average result over 1000 initial configurations. We denote this value as \bar{t}_i .

4 Experimental Results

We performed experiments on four patterns presented in Figure 3. They were denoted as pattern 1, pattern 2, pattern 3 and pattern 4, respectively. For each pattern, we tested the performance of the GA for three values of q : 0.1, 0.3 and 0.5. The maximal number of time steps t during which a CA has to converge to a desired final configuration was set to 100. Experiments showed that such a value is large enough to let good rules to converge to a desired final configuration. On the other hand, when a CA converged to a stable configuration earlier, the process of CA run was stopped.

The parameters of the GA were the following: $P = 200$, $t_{size} = 3$, $p_c = 0.7$ and $p_m = 0.02$. Higher than the usual mutation rate results from rather long chromosome and an enormous search space. Experiments show that slightly greater p_m helps the GA in the searching process. The searching phase was conducted through $G = 200$ generations. Increasing the number of generations had no effect on improving results.

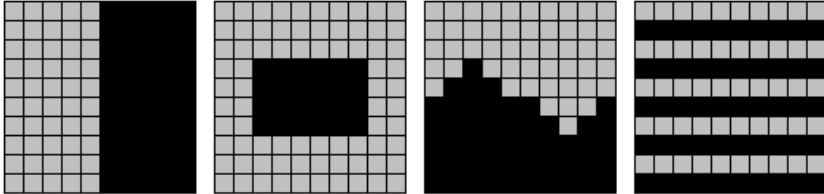


Fig. 3. Four patterns used in experiments: pattern 1, pattern 2, pattern 3, pattern 4 (from the left to the right)

As an example, figures 4 and 5 present typical cases of the GA run for pattern 2 and pattern 4. On each plot we can see the fitness value of the best individual in a given generation, for three values of q .

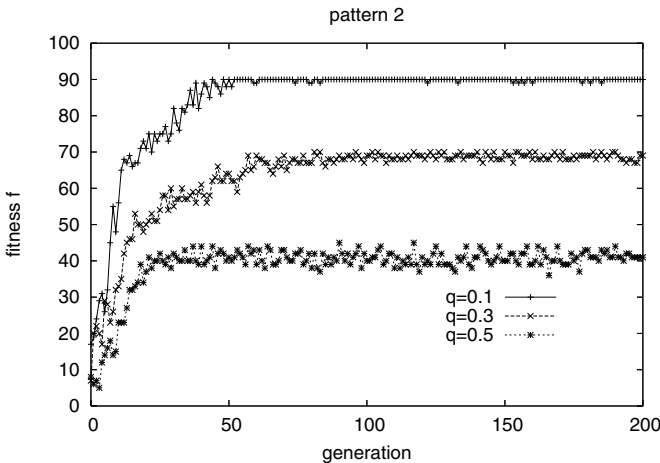


Fig. 4. The GA runs for pattern 2

The maximal fitness value for $q = 0.1$ equals to 90, for $q = 0.3$ equals to 70 and for $q = 0.5$ equals to 50. One can see that in case of $q = 0.1$, the GA is able to find a rule with the maximal fitness value. However, in some cases a fitness

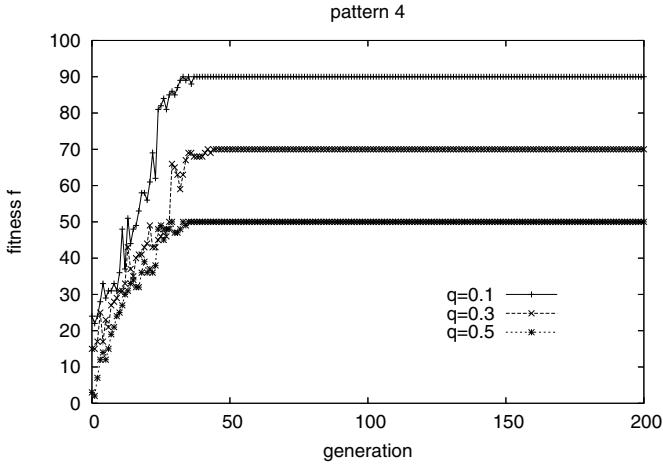


Fig. 5. The GA runs for pattern 4

value of the best individual slightly oscillates around its maximal value. The main reason is that the fitness function is stochastic: a rule might have different fitness values in each generation, depending on a concrete initial configuration.

As q increases, the quality of discovered rules usually slightly decreases, as was to be expected. Higher values of q mean more unknown states in an initial configuration (and less states known). In such cases, the CA rules may have not sufficient number of opportunities to be perfectly trained. For $q = 0.5$ the worst fitness value was observed for pattern 2. In last generations this value was oscillating between 40 and 43. On the other hand, rules discovered for pattern 4 seem to be perfect. In this case, the GA quickly discovers rules with the maximal fitness value. What is interesting is that for this pattern the value of q has no effect on quality of discovered rules.

The real quality of discovered rules is measured during the normal operating phase. For each pattern, we tested the final population discovered by the GA on 1000 random initial configurations. For each rule, \bar{t}_i was computed. The results of the best rules (from the whole population), denoted as \bar{t}_{best} are gathered in Table 1.

These data confirm results from the learning phase but they are more accurate since CA rules were tested on 1000 random initial configurations. Slight changes of these values may occur while another testing since initial configurations were created in a random way.

Let us look closely at the performance of the best rules from the normal operating phase. For example, let us take the best rule found after this phase for pattern 4 and for $q = 0.5$. This rule (the chromosome of the GA) is presented in Tab. 2. Figure 6 presents the initial configuration of the CA and further configurations in time steps: 1, 29 and 39.

Table 1. t_{best}^- values for different patterns and different q values

	q=0.1	q=0.3	q=0.5
pattern 1	0.995	0.960	0.914
pattern 2	0.971	0.918	0.760
pattern 3	0.966	0.925	0.885
pattern 4	1.0	1.0	1.0

Table 2. The chromosome of the best rule found for pattern 4 and for $q = 0.5$

222122121121211222122112221122211122211111112211211222122222111212222112222
 112122221121121122111221112112222122222121222221222111222221222112122222122
 12221212222222122122112112122112212111111122121112112121212221121112112222221
 111211211122111

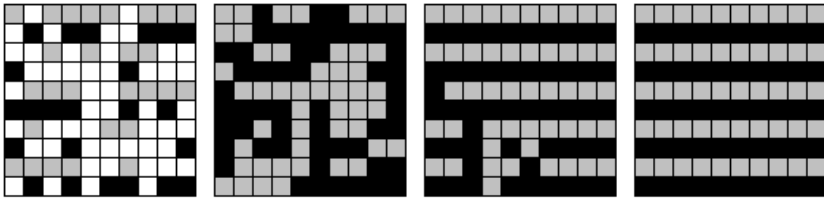


Fig. 6. Configurations of the CA in time steps: 0, 1, 29 and 39 (from the left to the right)

Since $q = 0.5$, the initial configuration has 50 cells of unknown state (state 0). In step 1 all cells with unknown state are eliminated. However, the configuration is not proper yet. Looking closely at Figure 6 one can see that in intermediate time steps, some cells which states are determined in the initial configuration, change their states. From the point of view of our task this situation seems to be undesirable. In spite of this, in subsequent time steps the CA rule corrects cells' states and finally, at time step 39, the CA converges to desired configuration. This final configuration is identical as pattern the 4, what means perfect reconstruction. This rule obtained t_{best}^- value equal to 1.0

Our previous work [10] concerning discovery by GA of CA rules, conducted for multiprocessor scheduling task, showed interesting possibilities of discovered rules. Namely, rules discovered for one problem can be successfully used for problems similar to a given one. Following those results we conducted a sequence of experiments in which rules discovered for one pattern were used to reconstruct another pattern, omitting the learning phase. Patterns used in these experiments are presented in Figure 7. These patterns, named as pattern 1', pattern 2', pattern 3' and pattern 4' were created on the base of patterns: 1, 2, 3 and 4, respectively. Each of these new patterns is in some way similar to its original.

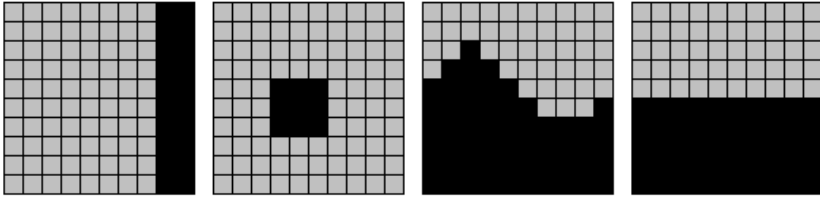


Fig. 7. Four patterns used in experiments: pattern 1', pattern 2', pattern 3', pattern 4' (from the left to the right)

In the normal operating phase we tested populations of rules discovered for four patterns from Figure 3 on their modified patterns presented in Figure 7. More precisely, rules discovered for pattern 1 were tested on not complete pattern 1', etc. In our experiments we used $q = 0.1$. The results of the best rules, denoted as t_{best} , for pattern 1', pattern 2', pattern 3' and pattern 4' were the following: 0.991, 0.971, 0.969 and 0.694, respectively. From these results one can conclude that some of the rules are more general (rules for patterns 1,2 and 3) and some are not (rules for pattern 4). This experiment confirms possibilities of discovered rules in solving new problems [10], which are similar to the previously solved.

5 Conclusions

In this paper we have presented a new task for CAs: pattern reconstruction task. The aim of the work was finding a CA rule which is able to transform an initial, not complete configuration to the final complete configuration. Results of presented experiments show that the GA is able to discover rules suitable to solve this task for a given instance of a problem. Found rules perform well even when the number of unknown cells is relatively high. The issue which must be examined closely is the influence of the neighborhood type and perhaps other kind of rules on quality of results.

Another subject is the scalability of discovered rules. How will rules discovered for a given grid size perform on larger grids? On the base of the CAs behavior one may expect that such scalability exists, but this must be confirm by experiments.

An important subject addressed in the paper is reusing discovered CA rules. Experiments show that during learning phase rules store some kind of knowledge about pattern which is reconstructed. This knowledge can be successfully reused in the process of reconstructing other patterns, which are similar to the previously reconstructed. Another issue is the term of similarity in the context of patterns. The idea of proper and thoughtful reusing of discovered rules will be the subject of our future research.

Since results of conducted experiments are encouraging, we also plan to apply our algorithm in similar, more practical and realistic problems.

References

1. Bandini, S., Vanneschi, L., Wuensche, A., Shehata, A.B.: A Neuro-Genetic Framework for Pattern Recognition in Complex Systems. *Fundamenta Informaticae* 87(2), 207–226 (2008)
2. Banham, M.R., Katsaggelos, A.K.: Digital image restoration. *IEEE Signal Processing Magazine* 14, 24–41 (1997)
3. Breukelaar, R., Back, T.: Evolving Transition Rules for Multi Dimensional Cellular Automata. In: Sloot, P.M.A., Chopard, B., Hoekstra, A.G. (eds.) ACRI 2004. LNCS, vol. 3305, pp. 182–191. Springer, Heidelberg (2004)
4. Das, R., Crutchfield, J.P., Mitchell, M.: Evolving globally synchronized cellular automata. In: *Proceedings of the Sixth International Conference on Genetic Algorithms*. Morgan Kaufmann, San Francisco (1995)
5. Fawcett, T.: Data mining with cellular automata. *ACM SIGKDD Explorations Newsletter* 10(1), 32–39 (2008)
6. Hernandez, G., Herrmann, H.: Cellular automata for elementary image enhancement. *Graphical Models and Image Processing* 58(1), 82–89 (1996)
7. Mitchell, M., Hraber, P.T., Crutchfield, J.P.: Revisiting the Edge of Chaos: Evolving Cellular Automata to Perform Computations. *Complex Systems* 7, 89–130 (1993)
8. Rosin, P.L.: Training Cellular Automata for Image Processing. *IEEE Transactions on Image Processing* 15(7), 2076–2087 (2006)
9. Slatnia, S., Batouche, M., Melkemi, K.E.: Evolutionary Cellular Automata Based-Approach for Edge Detection. In: Masulli, F., Mitra, S., Pasi, G. (eds.) WILF 2007. LNCS (LNAI), vol. 4578, pp. 404–411. Springer, Heidelberg (2007)
10. Swiecicka, A., Seredynski, F., Zomaya, A.Y.: Multiprocessor scheduling and rescheduling with use of cellular automata and artificial immune system support. *IEEE Transactions on Parallel and Distributed Systems* 17(3), 253–262 (2006)
11. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)

Addition of Recurrent Configurations in Chip Firing Games: Finding Minimal Recurrent Configurations with Markov Chains

Matthias Schulz

Karlsruhe Institute for Technology, Department for Computer Sciences,
Am Fasanengarten 5, 76128 Karlsruhe, Germany
schulz@ira.uka.de

Abstract. Generalizing the Abelian Sandpile Model by Bak, Tang and Wiesenfeld to general undirected graphs, one gets a variation of the Chip Firing Game introduced by Chung and Ellis in 2002, which still contains most of the nice algebraic properties of the Abelian Sandpile Model. Particularly the group structure of the recurrent configurations is retained.

Using a Markov Chain, we show how a pair consisting of one minimal recurrent configuration and one nearly minimal recurrent configuration can be constructed whose sum is the same as the sum of a given pair of recurrent configurations.

Computer simulations of this Markov Chain for the Abelian Sandpile Model suggest that the number of steps needed to reach a final pair usually is proportional to the width of the grid, but can become proportional to the square of the width if one chooses particular configurations.

Keywords: Chip Firing Games, Abelian Sandpile Model, Minimal Recurrent Configurations, Addition of Recurrent Configurations.

1 Introduction

The Abelian sandpile model, introduced by Bak, Tang and Wiesenfeld in 1987 [1] as a model to explain $\frac{1}{f}$ noise, was generalized by Chung and Ellis in 2002 [2] to undirected graphs as a variant of Chip Firing Games. Chip firing is a game played on an undirected graph, where a vertex v containing at least $\deg(v)$ can “fire” and give one of its chips to each of its adjacent vertices.

Chung’s and Ellis’ innovation was the idea of taking chips out of the system when they fall onto a special subset of vertices, which brings this model closer to the notion of the sandpile model where grains of sand fall off the edge of the grid.

As proven in [2], many of the algebraic properties Dhar found for the sandpile model in [3] can directly be transferred to these Chip Firing Games (CFGs), among them the fact that the set of recurrent configurations is a group.

The group operation consists of component wise adding two configurations and then letting vertices fire until no firings are possible anymore. We show for each pair of recurrent configurations c, d that there exists a minimal recurrent

configuration, i.e. a recurrent configuration from which no chip can be taken without getting a non-recurrent configuration, and an almost minimal recurrent configuration such that the sum of these configurations is the same as the sum of c and d , and that there are fewer firings until a stable configuration is reached.

First, we will introduce the basic concepts for Chip Firing Games, before examining the relation between minimal recurrent configurations and the sequence in which the vertices fire for recurrent configurations.

We then define and theoretically analyze a Markov Chain before giving results obtained when simulated this Markov Chain for sandpile models of different sizes.

2 Preliminaries

2.1 Basic Definitions

An undirected graph $U = (V \cup S, E)$ is called a *CFG-graph* iff V and S are disjoint, each vertex $v \in S$ is adjacent to exactly one vertex which lies in V and there exists a path from each vertex $v \in V$ to a vertex $s \in S$.

A Chip Firing Game (CFG) on a CFG-graph defines a transition rule for configurations $c : V \rightarrow \mathbb{N}_0$:

If a vertex $v \in V$ satisfies $c(v) \geq \deg(v)$, v is called a *critical vertex* and we can let v "fire" and get a new configuration $\tau_v(c) : V \rightarrow \mathbb{N}_0$ defined through

$$\forall v' \in V : \tau_v(v') = \begin{cases} c(v') - \deg(v') & \text{if } v' = v \\ c(v') + 1 & \text{if } \{v, v'\} \in E \\ c(v') & \text{otherwise.} \end{cases}$$

We can interpret a configuration c as assigning each vertex a number of chips; if a vertex v fires, it gives $\deg(v)$ chips to its neighbors, which means one chip to each adjacent vertex.

Note that chips which are given to vertices in S simply vanish from the game.

A configuration which contains a vertex which is able to fire is called *critical*; a configuration which is not critical is called *stable*, and the set of stable configurations is denoted \mathcal{C}^U .

2.2 Relaxations of Configurations

It has been shown (for example in [2]) that, starting from a critical configuration, after a finite number of firings of critical vertices we reach a stable configuration. We call the process of these firings the *relaxation of c* and the sequence of vertices which fired the *firing sequence of c* .

It is also shown in [2] and [3] that the stable configuration reached does not depend on the sequence of firings - there exists a unique stable configuration c_{rel} a given critical configuration c relaxes into.

What is more, the number of times a given vertex fires during the relaxation of a critical configuration also only depends upon the critical configuration in question,

as also shown in [2]; the vector which assigns each vertex the number of times it fires during the relaxation of c is called the *firing vector of c* , denoted f_c .

Throughout this paper, when comparing different firing vectors or different configurations, we will use the relation \leq defined through $c \leq d \iff \forall v \in V : c(v) \leq d(v)$.

2.3 The Operation \oplus and Recurrent Configurations

We define the operation \oplus on \mathcal{C}^U through

$$\forall c, d \in \mathcal{C}^U : c \oplus d = (c + d)_{rel} = c + d - Bf_{c+d}.$$

(The operation $+$ is the usual pointwise addition of functions.)

Then \oplus is commutative and associative.

The introduction of the operation \oplus also allows us to state a useful fact about the firings when we consider the firing vectors of sums of three configurations:

$$\forall c, d, e \in \mathbb{N}^V : f_{c+d+e} = f_{d+e} + f_{c+(d \oplus e)},$$

which was proven in [5].

In other words, if one lets the sum $c + d + e$ relax, the procedure can be seen as first relaxing $d + e$ and ignoring c until one gets $c + (d \oplus e)$, which then gets relaxed until $c \oplus d \oplus e$ is reached.

A very nice property of the operation \oplus is that there exists a subset R of configurations in \mathcal{C}^U such that (R, \oplus) is an Abelian group.

The largest subset with this quality is the *set of recurrent configurations on U* , denoted \mathcal{R}^U .

From now on, let $b \in \mathcal{C}^U$ be the configuration which assigns to each vertex v the number of vertices in S which are adjacent to v . The configuration b is also called the *burning configuration of U* .

Generalizing the Dhar's Burning Algorithm from [4], we get the following statement which makes it easy to find out whether a given configuration c is recurrent or not:

$\forall c \in \mathcal{C}^U : c \in \mathcal{R}^U \iff$ the firing sequence for $c + b$ contains each vertex exactly once.

(Note that for all $c \in \mathcal{C}^U$ the firing sequence of $c + b$ contains each vertex at most once.)

For the rest of this paper, we will say that a sequence F of vertices is a firing sequence for a recurrent configuration c if F is the firing sequence for $c + b$.

We will take a good look at such firing sequences containing each vertex exactly once in the next section.

3 Firing Sequences and Minimal Recurrent Configurations

We say a recurrent configuration c is a *minimal recurrent configuration* iff $\forall v \in V : c - e_v \notin \mathcal{R}^U$ is true, i.e. if no chip can be taken away from the configuration

without the result not being a recurrent configuration, and denote the set of all minimal recurrent configurations \mathcal{R}_{min}^U .

For a sequence $F = (v_0, \dots, v_{|V|-1})$ which contains all vertices in V exactly once we define the configuration c_F through $\forall v \in V : c_F(v)$ is the number of vertices adjacent to v which come after v in F .

3.1 Relation between Firing Sequences and Minimal Recurrent Configurations

Let $F = (v_0, \dots, v_{|V|-1})$ be a sequence of vertices which contains all vertices in V .

Then $c_F \in \mathcal{C} \Rightarrow c_F \in \mathcal{R}_{min}^U$ is true and F is a firing sequence of c_F .

Proof: In this proof, we will call the vertices adjacent to v which come before/after v in F the F -predecessors/successors of v .

First, we show that c_F is a recurrent configuration if $c \in \mathcal{C}^U$.

For $v \in V$, we define $n_+(v)$ as the number of F -predecessors of v , n_-V as the number of F -successors of v and $n_s(v)$ as the number of neighbors v has in S .

Then $\text{deg}(v) = n_+(v) + n_-(v) + n_s(v)$ holds.

Consider the configuration $d = c_F + b$. According to definition, the number of chips on a vertex v is $n_+(v) + n_s(v)$.

This means that v still needs exactly $n_-(v)$ chips to become critical.

This means that F is a firing sequence for d : By the time the firing sequence says a vertex v should fire, n_- neighbors of v already will have fired, meaning that v has become critical at that point and can fire.

As $c_F \in \mathcal{C}^U$ and because all vertices fire during the relaxation of $c_F + b$, $c_F \in \mathbb{R}$ holds.

We now show that for all $v \in V$ the configuration $c_F - e_v$ cannot lie in \mathcal{R}^U , which is obvious if $c_F(v) = 0$.

If $c_F(v) > 0$, we consider the set N of all vertices $u \in V$ for which a path $(v = v_0, \dots, v_k = u)$ exists, such that v_{i+1} always is a F -successor to v_i .

Note that $v \in N$ when one considers a path of length zero, and that if a vertex u lies in N , then all F -successors of u also lie in N .

Suppose that a vertex in N can fire during the relaxation of $d = c_F - e_v + b$. The vertex u shall be then be the first vertex in N to fire. As all F -successors of u also lie in N , none of these neighbors can fire before u fires.

If $u = v$ there have to be $n_- + 1$ neighbors of v which fire before v . As there are only n_- neighbors which do not lie in N , this is not possible.

If $u \neq v$, we consider the path $(v = v_0, \dots, v_k = u)$ as described above. This means that all vertices of the path lie in N , and especially v_{k-1} which is an F -predecessor to u .

Therefore u can have at most $n_- - 1$ neighbors which do not lie in N , and therefore cannot become critical during the relaxation of $c_F - e_v + b$.

As there are vertices which cannot fire during the relaxation of $c_F - e_v + b$, $c_F - e_v$ cannot be recurrent.

If $c_F \in \mathcal{C}^U$ we therefore get $c_F \in \mathcal{R}_{min}^U$. □

Note that we also found that each vertex v contains exactly $\deg(v)$ chips when it is its turn to fire during the relaxation; this is true for all minimal recurrent configurations.

3.2 Switching Chips in Minimal Recurrent Configurations

If $c \in \mathcal{R}_{min}^U$ is a minimal recurrent configuration and the vertex $v \in V$ satisfies $c(v) + b(v) < \deg(v)$ and $c(v) < \deg(v) - 1$, then there exists a vertex $v' \in V$ with $\{v, v'\} \in E$ such that $c + e_v - e_{v'} \in \mathcal{R}_{min}^U$ holds.

In other words, one can take one chip off v' and add it to v and still gets a minimal recurrent configuration as a result.

The idea is to take the firing sequence $F = (v_0, \dots, v_{|V|-1})$ which contains $v = v_i$ for some i , choose $v_j = v'$ as the last vertex in F with $\{v, v'\} \in E$ which comes before v in F and then set $F' = (v_0, \dots, v_i, v_j, v_{j+1}, \dots, v_{i-1}, v_{i+1}, v_{i+2}, \dots, v_{|V|-1})$.

It is easy to verify that F' is the firing sequence for $c + e_v - e_{v'}$, which proves the claim.

4 Markov Chain on Triples of Configurations

We define the set $T = \mathcal{R}_{min}^U \times \mathcal{R}_{min}^U \times \mathcal{C}^U$ and the relation \rightarrow on the set T by

$$\forall \gamma_1 = (c_1, c_2, d), \gamma_2 = (c'_1, c'_2, d) \in T : \gamma_1 \rightarrow \gamma_2 \iff c'_2 = c_1 \wedge c_2 \oplus d = c'_1 + d'$$

The relation \rightarrow^* is the reflexive transitive closure of \rightarrow .

Note that, as a firing sequence for a recurrent configuration usually is not unique, there can be different successors to a given triple $\gamma \in T$.

For all triples $\gamma = (c_1, c_2, d), \gamma' = (c'_1, c'_2, d') \in T$ with $\gamma \rightarrow \gamma'$ we can show that $c_1 \oplus c_2 \oplus d = c'_1 \oplus c'_2 \oplus d'$ and $f_{c_1+c_2+d} = f_{c'_1+c'_2+d'} + f_{c_2+d}$ (and therefore $f_{\gamma'} = f_{c'_1+c'_2+d'} \leq f_{c_1+c_2+d} = f_{\gamma}$) is true.

In other words, the number of firings the sum of the three configurations causes decreases while the relaxed sum stays the same.

The claims follow directly from the fact that $c'_1 + c'_2 + d' = c_1 + (c_2 \oplus d)$.

4.1 Recurrent Triples

We now define a Markov Chain with elements in T with a starting triple $\gamma_0 = (c_1^0, c_2^0, d^0)$, where the successor to each element γ is one of the triples γ' satisfying $\gamma \rightarrow \gamma'$, and consider the set of recurrent elements of this Markov Chain, called *recurrent triples*.

All recurrent elements $\gamma \in T$ of the Markov Chain which are eventually reached satisfy $\forall \gamma' \in T : \gamma \rightarrow^* \gamma' \Rightarrow f_{\gamma} = f_{\gamma'}$, as f_{γ} is monotonically decreasing and therefore $\gamma' \rightarrow \gamma$ would not be possible if $f_{\gamma} \neq f_{\gamma'}$ would be true.

It follows that for a recurrent triple (c_1, c_2, d) $c_2 + d$ is a stable configuration, since the firing vector for any successor would decrease otherwise.

4.2 Final Triples

From now on, the subgraph of the undirected CFG-graph $U = (V \cup S, E)$ induced by V shall be bipartit, with $V_0, V_1 \subseteq V$ being subsets which satisfy

$$V_0 \cap V_1 = \emptyset, V_0 \cup V_1 = V \wedge \forall \{u, v\} \in E : u \in V_0 \iff v \in V_1.$$

We define the configuration $b' \in \mathcal{C}$ through $\forall v \in V : b'(v) = \max(b(v) - 1, 0)$ and outline how to show that for a recurrent triple $(c_1, c_2, d) \in T$ there exist a triple $(c'_1, c'_2, d') \in T$ with $(c_1, c_2, d) \rightarrow^* (c'_1, c'_2, d')$ satisfying $d' \leq b'$.

(We will call triples $(c'_1, c'_2, d') \in T$ satisfying $d' \leq b'$ *final triples*.)

First, one shows that if $(c_1, c_2, e_v) \in T$ is a recurrent triple and one always chooses a successor to $(c'_1, c'_2, e_{v'})$ to a triple $(\bar{c}_1, \bar{c}'_2, \bar{e}_{v'})$ in such a way that v and v' are adjacent, one eventually reaches a triple $(\tilde{c}_1, \tilde{c}_2, e_u)$ for which this is not possible. To do so, one shows that the function $\Theta : \mathbb{N}_0 \rightarrow \mathbb{N}_0, i \mapsto \sum_{v \in V_{i \bmod 2}} c_1^i(v) + \sum_{v \in V_{(i+1) \bmod 2}} c_2^i(v)$ decreases in each step (this is where it is important that U is a bipartite graph); as $\Theta(i)$ can never be negative there can only be finite many steps before one cannot find a successor with the given property.

Subsection 3.2 and the fact that the sum of the last two components of the last triple $(\tilde{c}_1, \tilde{c}_2, e_u)$ do not cause any firings imply that $b(u) \geq 2$ and $\tilde{c}_2(u) = \deg(u) - b(u)$.

It is easy to verify that for $d_1 \leq d_2$ the implication

$$(c_1, c_2, d_1) \rightarrow^* (c'_1, c'_2, d') \Rightarrow (c_1, c_2, d_2) \rightarrow^* (c'_1, c'_2, d' + (d_2 - d_1))$$

is always true if (c_1, c_2, d_2) is recurrent.

Lastly, we do an induction over the number k of chips d contains to show that we can reach a final triple from each recurrent triple (c_1, c_2, d) :

If $k = 0$, (c_1, c_2, d) already is a final triple.

If $k \geq 1$, we can take a configuration $d' \leq d$ containing k chips and a vertex $v \in V$ such that $d = d' + e_v$.

Our induction hypothesis tells us we can reach a final triple (c'_1, c'_2, d'') from (c_1, c_2, d') , and therefore $(c'_1, c'_2, d'' + e_v)$ is reachable from (c_1, c_2, d) .

From (c'_1, c'_2, e_v) we can reach a final triple $(\bar{c}_1, \bar{c}_2, e_{v'})$, and therefore $(\bar{c}_1, \bar{c}_2, e_{v'} + d'')$ is reachable from $(c'_1, c'_2, d'' + e_v)$ and therefore also from (c_1, c_2, d) .

Using the fact that for $u \neq v'$ $(d'' + e_{v'})(u) = d''(u) \leq b'(u)$ holds according to our induction hypothesis, that $\bar{c}_2(v') = \deg(v') - b(v')$ and that $c_2(v') + d''(v') + e_{v'} + v' \leq \deg(v')$ holds to show that $d'' + e_{v'} \leq b'(v)$ is true and we reached a final triple.

4.3 Corollaries

- a) For all recurrent configurations $c_1, c_2 \in \mathcal{R}^U$ there exist two minimal recurrent configurations $c'_1, c'_2 \in \mathcal{R}_{min}^U$ and a configuration $d' \in \mathcal{C}, d' \leq b'$ such that $c'_1 \oplus c'_2 \oplus d' = c \oplus d \wedge f_{c'_1+c'_2+d'} \leq f_{c+d}$ gilt.

In other words, we can always find a minimal recurrent configuration and a recurrent configuration which is "almost" a minimal recurrent configuration whose sum relaxes to $c \oplus d$ with at most as many firings as there are during the relaxation of $c + d$.

This triple is a final triple one can reach from the starting triple $(c'_1, c'_2, d) \in T$ with $c'_1 \leq c_1, c'_2 \leq c_2$ and $d = (c_1 - c'_1) \oplus (c_2 - c'_2)$. \square

- b) Let $c \in \mathcal{R}^U$ be a recurrent configuration and $S_C = \{(c_1, c_2) \in \mathcal{R}^U \times \mathcal{R}^U \mid c_1 \oplus c_2 = c\}$ the set of all pairs of recurrent configurations whose sum relaxes to c .

There exist two minimal recurrent configurations $m_1, m_2 \in \mathcal{R}_{min}^U$ as well as a configuration $d \leq b'$ such that $(m_1, m_2 + d) \in S_c$ and the number of firings during the relaxation of $m_1 + (m_2 + d)$ is minimal among all pairs $(c_1, c_2) \in S_c$.

Note that if $\forall v \in V : b(v) \leq 1$, the configuration d is always 0 and we get the statement that there are two minimal recurrent configurations $m_1, m_2 \in \mathcal{R}_{min}^U$ such that $(m_1, m_2) \in S_c$ and the number of firings during the relaxation of $m_1 + m_2$ is minimal for all pairs in S_c .

This means that it is sensible to look for the pair (d_1, d_2) whose relaxed sum is a given recurrent configuration c and whose sum's relaxation contains as few firings as possible among all pairs in S_c among the set of all pairs of minimal recurrent configurations whose relaxed sum is c (or $c - d'$ for different configurations $d' \leq b'$ if there exists a vertex $v \in V$ with $b(v) \geq 2$).

We can prove this by using Corollary [a\)](#) for a pair $(c_1, c_2) \in S_c$ for which the number of firings is minimal. \square

5 Time Complexity

In this section, we look at the number of steps needed to reach a final triple $(c'_1, c'_2, d') \in T$ when starting with an initial triple (c_1, c_2, d) .

Therefore, we have to specify how we compute successors of a triple $(c_1, c_2, d) \in T$, which leads to an implicit assignment of transition probabilities for the Markov Chain.

5.1 Choosing Random Firing Sequences

Given a recurrent configuration $c \in \mathcal{R}^U$, we compute a firing sequence for c as follows:

We set the configuration $d_0 = c + b$ and create a list L_0 of all vertices in V satisfying $d_0(v) \geq \text{deg}(v)$.

As long as d_i is critical, we randomly choose a vertex v_i from L_i (all vertices shall be chosen with the same probability) and let v_i fire to get d_{i+1} , with L_{i+1} being the list of all vertices critical in d_{i+1} .

Note that each firing sequence has a probability > 0 of being chosen in the end, although not all firing sequences necessarily have the same probability of being chosen.

5.2 Choosing Direct Successors

Given a triple $(c_1, c_2, d) \in T$, we compute the direct successor $(\bar{c}_1, \bar{c}_2, \bar{d})$ as follows:

- First, we set $\bar{c}_2 = c_1$.
- Then we compute $c_2 \oplus d$ and randomly choose a firing sequence F of $c_2 \oplus d$.
- We set $\bar{c}_1 = c_F$ and $\bar{d} = (c_2 \oplus d) - c_F$.

Note that all triples $(\bar{c}_1, \bar{c}_2, \bar{d})$ with $(c_1, c_2, d) \rightarrow (\bar{c}_1, \bar{c}_2, \bar{d})$ have a non-zero possibility of being chosen, although not all possible successors are equally likely to be chosen.

For the computations, the graph $U = (V \cup S, E)$ will always be a square grid with S consisting of vertices added to the vertices on the edges of the square (which means that vertices in the corners of the grid have two adjacent vertices in S). We say U is an $n \times n$ grid if the subgraph of U induced by V is an $n \times n$ grid, i.e. we ignore S .

We will look at different sizes of the grid and how the number of successors one has to compute before getting a final triple (c'_1, c'_2, d') . At first, we will randomly initiate the starting triple $(c_1, c_2, d) \in T$; then we will consider a special case, where the number of steps is far higher than for random starting triples.

5.3 Random Initial Triples

We randomly choose a firing sequence for the configuration $m \in \mathcal{R}^U$ satisfying $\forall v \in V : m(v) = \deg(v) - 1 = 3$. Then every firing sequence of a recurrent configuration is also a firing sequence of m , which means that all firing sequences have a non-zero probability of being chosen.

We set $c_1 = c_F$.

Next, we add a number k of chips to the empty configuration, each one of them is added to a vertex randomly chosen, with all vertices of V having the same chance of being chosen.

If k was chosen large enough, the relaxed configuration c' will be recurrent, and we will choose c_2 and d in such a way that $c_2 + d = c'$ is true.

To get the configurations c_2 and d , we randomly choose a firing sequence F' for c' and set $c_2 = c_{F'}$ and $d = c' - c_2$.

We computed the successors of the triple (c_1, c_2, d) for $n \times n$ grids with $n = 16, 32, 64, 128, 256$, adding $k = 2500, 10000, 40000, 160000, 640000$ chips to the empty configuration to get c' .

For each size, we had ten runs, always with a new starting triple. Table [1](#) gives the encountered maxima, minima and average number of successors computed before reaching a final triple.

The numbers suggest that the average number of steps needed is proportional to the width of the square grid, with a constant factor between 1.5 and 1.6.

Table 1. Number of steps needed to find a final triple, starting with random triples

n	Minimal number of steps	Maximal number of steps	Average number of steps
16	15	34	26.1
32	36	77	48.3
64	76	154	99.4
128	166	269	199.5
256	329	497	393.7

5.4 Particular Triples

As U is an $n \times n$ grid, we address each vertex v by its coordinates (x_v, y_v) .

$c_1(c_2)$ is the configuration which assigns each vertex exactly two chips, except the vertex in the upper left (lower right) corner which has no chips. All other vertices on the upper (lower) as well as the left (right) edge contain one chip.

We set $d = e_{(n \div 2, n \div 2)}$.

If we look at the firing sequences for c_1 and c_2 , we see that if a vertex u comes before a neighbor vertex v in one firing sequence, u comes after v in the other firing sequence. This leads to the chip contained in the last component going alternately in opposite directions, which means that it takes more steps for that one chip to reach the edge (where it either vanishes or goes into a corner) than it does for the many chips that vanished when we ran simulations for the random starting triple.

Table 2 lists the number of steps needed to get to a final triple when starting from (c_1, c_2, d) as defined above for $n = 16, 32, 64, 128, 256$, again looking at the results of ten runs each.

Table 2. Number of steps needed to find a final triple, starting with defined (c_1, c_2, d)

n	Minimal number of steps	Maximal number of steps	Average number of steps
16	60	371	207.7
32	675	1118	895.1
64	2857	3753	3222.4
128	11102	14566	12942.2
256	43912	52231	48278.9

The numbers in Table 2 suggest that the average number of steps needed to reach a final triple is proportional to n^2 with the constant factor being between 0.73 and 0.87. (However, it can be shown that actually two steps would be sufficient to get to a final triple.)

6 Conclusion

We have defined a Markov Chain whose elements are triples consisting of two minimal recurrent configurations and a third configuration of "surplus" chips. We

found that the number of firings needed to relax the sum of those configurations decreases as we progress in the Markov Chain, and we have been able to prove that, if the underlying graph is bipartite, from each triple we can reach a triple whose third component is bounded by a configuration containing even fewer chips than the burning configuration.

Computer simulations of the Markov Chain on an $n \times n$ grid suggest that usually the average number of steps of the Markov Chain lies in $O(n)$, ; however, we have also found starting triples where the average number of steps needed to reach a final triple seems to lie in $\Theta(n^2)$; these, however, are conjectures which should be dealt with more rigorously in future work.

If we consider configurations $c \in \mathcal{R}^U$ for which a minimal recurrent configuration $c' \in \mathcal{R}_{min}^U$ and a configuration $d \leq b'$ exist such that $c = c' + d$ holds as *nearly minimal recurrent configurations*, we can say that amongst the pairs of recurrent configurations whose relaxed sum is a constant configuration e , the fewest firings during the relaxation will always occur when the pair consists of a minimal recurrent configuration and a nearly minimal configuration, which usually seem to be computable in an acceptable time.

Starting from these, techniques to search for better pairs of recurrent configurations, i.e. pairs whose sum causes even fewer firings during the relaxation, can be developed; the structures of these pairs of configurations also is possibly helpful for the analysis of the question whether the problem to decide whether a sum of two recurrent configurations can cause fewer than a given number of firings.

Lastly, it would be of interest either to generalize the findings about the Markov Chains to graphs which are not bipartite or to find a graph and a starting triple such that no final triple as defined in this paper can be reached.

References

1. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality: An explanation of the $1/f$ noise. *Phys. Rev. Lett.* 59, 381–384 (1987)
2. Chung, F., Ellis, R.: A chip-firing game and dirichlet eigenvalues. *Discrete Mathematics* 257, 341–355 (2002)
3. Dhar, D., Ruelle, P., Sen, S., Verma, D.N.: Algebraic aspects of abelian sandpile models. *J. Phys. A* 28, 805 (1995)
4. Majumdar, S.N., Dhar, D.: Equivalence between the abelian sandpile model and the $q \rightarrow 0$ limit of the potts model. *Physica A: Statistical and Theoretical Physics* 185, 129–145 (1992)
5. Schulz, M.: On the addition of recurrent configurations of the sandpile-model. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008. LNCS*, vol. 5191, pp. 236–243. Springer, Heidelberg (2008)

A Seven-State Time-Optimum Square Synchronizer

Hiroshi Umeo* and Keisuke Kubo

Univ. of Osaka Electro-Communication,
Neyagawa-shi, Hastu-cho, 18-8, Osaka, 572-8530, Japan
umeo@cyt.osakac.ac.jp

Abstract. The firing squad synchronization problem on cellular automata has been studied extensively for more than fifty years, and a rich variety of synchronization algorithms have been proposed for not only one-dimensional arrays but two-dimensional arrays. In the present paper, we propose a seven-state optimum-time synchronization algorithm that can synchronize any square arrays of size $n \times n$ with a general at one corner in $2n - 2$ steps, which is a smallest realization of time-optimum square synchronizer known at present. The implementation is based on a new, simple zebra-like mapping scheme which embeds synchronization operations on one-dimensional arrays onto square arrays.

1 Introduction

We study a synchronization problem that gives a finite-state protocol for synchronizing a large scale of cellular automata. The synchronization in cellular automata has been known as a firing squad synchronization problem (FSSP) since its development, in which it was originally proposed by J. Myhill in Moore [1964] to synchronize all parts of self-reproducing cellular automata. The problem has been studied extensively for more than fifty years [1-15].

In the present paper, we propose a seven-state optimum-time synchronization algorithm that can synchronize any square arrays of size $n \times n$ with a general at one corner in $2n - 2$ steps, which is a smallest realization known at present. The implementation is based on a new, simple zebra mapping scheme which embeds synchronization operations on one-dimensional arrays onto square arrays. Not only the number of states of the implementation is small, but the correctness of the constructed transition function with 787 rules is clear and transparent.

2 Firing Squad Synchronization Problem on Two-Dimensional Square Arrays

Figure 1 shows a finite two-dimensional (2-D) square array consisting of $n \times n$ cells. Each cell is an identical (except the border cells) finite-state automaton. The array operates in lock-step mode in such a way that the next state of each

* Corresponding author.

cell (except border cells) is determined by both its own present state and the present states of its north, south, east and west neighbors. Thus, we assume the von Neumann-type four nearest neighbors. All cells (*soldiers*), except the north-west corner cell (*general*), are initially in the quiescent state at time $t = 0$ with the property that the next state of a quiescent cell with quiescent neighbors is the quiescent state again. At time $t = 0$, the north-west corner cell C_{11} is in the *fire-when-ready* state, which is the initiation signal for synchronizing the array. The firing squad synchronization problem is to determine a description (state set and next-state function) for cells that ensures all cells enter the *fire* state at exactly the same time and for the first time. The tricky part of the problem is that the same kind of soldier having a fixed number of states must be synchronized, regardless of the size $n \times n$ of the array. The set of states and next state function must be independent of n .

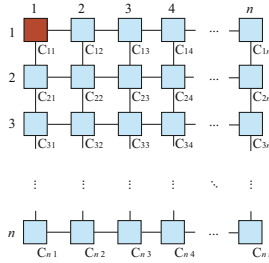


Fig. 1. A two-dimensional square cellular automaton

The problem was first solved by J. McCarthy and M. Minsky who presented a $3n$ -step algorithm for one-dimensional cellular array of length n . In 1962, the first optimum-time, i.e. $(2n - 2)$ -step, synchronization algorithm was presented by Goto [1962], with each cell having several thousands of states. Mazoyer [1987] developed a six-state synchronization algorithm which, at present, is the algorithm having the fewest states for one-dimensional arrays. On the other hand, a rich variety of synchronization algorithms for two-dimensional rectangle arrays have been proposed [5-8, 10-14]. As for square synchronization which is a special class of rectangles, several square synchronization algorithms have been proposed by Beyer [1969], Shinahr [1974], and Umeo, Maeda, and Fujiwara [2002]. The first optimum-time square synchronization algorithm was proposed by Beyer [1969] and Shinahr [1974]. One can easily see that it takes $2n - 2$ steps for any signal to travel from C_{11} to C_{nn} due to the neighborhood size. Concerning the time optimality of the two-dimensional square synchronization algorithms, the following theorems have been shown.

Theorem 1. Beyer [1969], Shinahr [1974] There exists no cellular automaton that can synchronize any two-dimensional square array of size $n \times n$ in less than $2n - 2$ steps, where the general is located at one corner of the array.

Theorem 2. ^{Shinahr [1974]} There exists a 17-state cellular automaton that can synchronize any two-dimensional square array of size $n \times n$ at exactly $2n - 2$ optimum steps, where the general is located at one corner of the array.

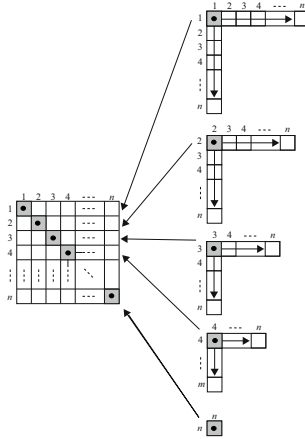


Fig. 2. A synchronization scheme for $n \times n$ square cellular automaton. A horizontal and vertical synchronization operations on L_i are mapped onto a square array. A black circle \bullet in a shaded square represents a general on each L_i and a wake-up signal for the synchronization generated by the general is indicated by a horizontal and vertical arrow.

The optimum-time synchronization algorithm for square arrays operates as follows: We assume that an initial general is located on C_{11} . By dividing the entire square array of size $n \times n$ into n rotated *L-shaped* 1-D arrays, shown in Fig. 2, in such a way that the length of the i th (from outside) L-shaped array is $2n - 2i + 1$ ($1 \leq i \leq n$), one treats the square synchronization as n independent 1-D synchronizations with the general located at the bending cell of the L-shaped array. We denote the i th L-shaped array by L_i and its horizontal and vertical segment is denoted by L_i^h and L_i^v , respectively. Note that a cell at each bending point of the L-shaped array is shared by the two segments. See Fig. 2.

Concerning the synchronization of L_i , it can be easily seen that a general is generated at the cell C_{ii} at time $t = 2i - 2$ with the four nearest von-Neumann neighborhood communication, and the general initiates the horizontal (row) and vertical (column) synchronizations on L_i^h and L_i^v , each of length $n - i + 1$ via an optimum-time synchronization algorithm which can synchronize arrays of length ℓ in $2\ell - 2$ steps. Thus the square array of size $n \times n$ can be synchronized at time $t = 2i - 2 + 2(n - i + 1) - 2 = 2n - 2$ in optimum-steps. In Fig. 2, each general is represented by a black circle \bullet in a shaded square and a wake-up signal for the synchronization generated by the general is indicated by a horizontal and vertical arrow.

The algorithm itself is very simple and now we are going to discuss its implementation in terms of a 2-D cellular automaton. The question is: how many states are required for its realization? Let Q be a set of internal states for the 1-D optimum-time synchronization algorithm which is embedded as a base algorithm. When we implement the algorithm on square arrays based on the scheme above, we usually prepare a different state set used by the cells on L_i^h and L_i^l , which is in the upper and lower triangle areas separated by the principal diagonal. Thus, $2 \| Q \| - 1$ states are usually required for its independent row and column synchronization operations in order to avoid state mixing. Only a firing state is shared by the two areas. Shinahr [1974] gave a 17-state implementation based on Balzer's eight-state synchronization algorithm (Balzer [1967]). Later, it has been shown in Umeo, Maeda and Fujiwara [2002] that nine states are sufficient for the optimum-time square synchronization:

Theorem 3. Umeo et al. [2002] There exists a 9-state 2-D CA that can synchronize any $n \times n$ square array in $2n - 2$ steps.

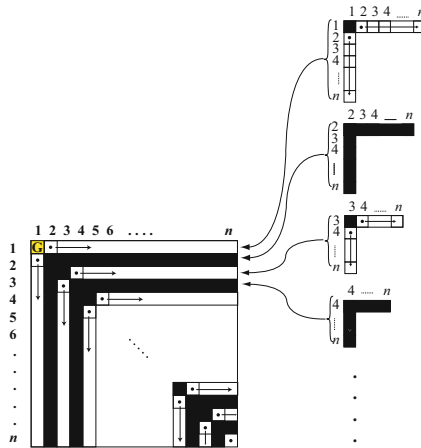


Fig. 3. A zebra mapping scheme for $n \times n$ square cellular automaton

Umeo et al. [2002] constructed the nine-state implementation by applying the Mazoyer's 6-state one-dimensional synchronization algorithm (Mazoyer [1987]). It has been shown that two more states can be deleted from the usual construction above, rendering nine states sufficient for optimum-time square synchronization. We have tested the transition rule set for its validity on square arrays of size 2×2 to 1000×1000 on a computer, however, showing the correctness of the 9-state implementation was very difficult due to its tricky construction. The main objective of this paper is to explore a state-efficient, i.e., small number of states, implementation that holds an easy way of showing its correctness.

3 A Seven-State Time-Optimum Square Synchronizer

In this section we propose an optimum-time square synchronization algorithm \mathcal{A} and its seven-state implementation is provided. We show that seven states are sufficient for the optimum-time square synchronization. The correctness of the constructed transition rule set is clear from the simple construction.

3.1 Zebra Mapping

The proposed algorithm is basically based on the rotated L-shaped mapping scheme presented in the previous section, however, the mapping onto square arrays consists of two types of configurations: one is a one-cell smaller synchronized configuration and the other is a filled-in configuration with a stationary state. The stationary state remains unchanged once filled-in by the time before the final synchronization. Each configuration is mapped alternatively onto an L-shaped array in a zebra fashion. The mapping is referred to as zebra mapping. Figure 3 illustrates the zebra mapping which consists of an embedded synchronization and a filled-in layer.

A key idea of our small-state implementation is:

- Alternative mapping of two types of configurations. A stationary layer separates two consecutive synchronization layers and it allows us to use a same state set for the vertical and horizontal synchronization on each layer, helping us to construct a small-state transition rule set for the synchronization layers.
- A one-cell smaller synchronization configuration than previous is embedded, where we can save synchronization time by two steps.
- A single state X is shared between an initial general state of the square synchronizer, the stationary state, and a firing state of the embedded one-dimensional synchronization algorithm used. The state X itself acts as a pre-firing state.
- Any cell in state X , except $C_{n,n}$, enters the synchronization state at the next step if all its neighbors are in state X or the boundary state of the square. The cell $C_{n,n}$ enters the synchronization state if and only if its north and west cells are in state X and its east and south cells are in the boundary state. A cell in state X that is adjacent to the cell $C_{n,n}$ is also an exception. This is an only condition that makes cells fire.

3.2 Seven-State Implementation

In our construction we take the Mazoyer's 6-state one-dimensional synchronization rule as an embedded synchronization algorithm. The set of the 6-states is $\{G, Q, A, B, C, X\}$, where G is a general, Q is a quiescent, and X is a firing state, respectively. The other three states A , B and C are auxiliary states, respectively.

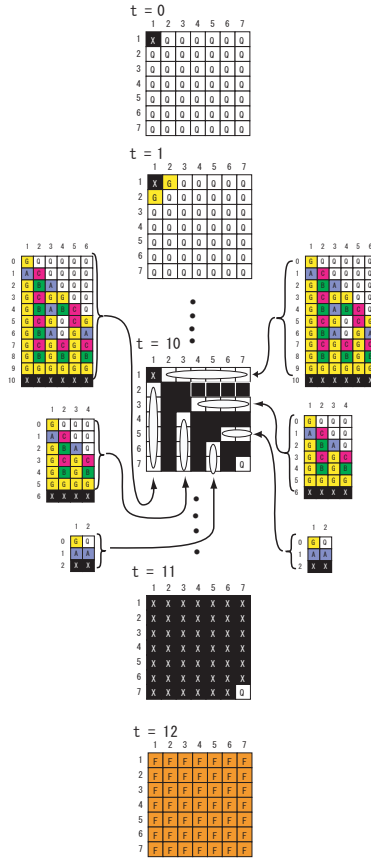


Fig. 4. A zebra implementation of six-state Mazoyer’s rule on 7×7 square cellular automaton. Configurations of length 6, 4 and 2 are mapped on the odd synchronization layers.

The seven-state square synchronizer that we construct has the following state set: $\{G, Q, A, B, C, X, F\}$, where F is a newly introduced firing state, X is a general, and Q is a quiescent state, respectively. The state G is the general state of the embedded synchronization. Those states A, B and C are also auxiliary states, respectively. The transition rule set is constructed in such a way that: The initial general on $C_{1,1}$ in state X generates a new general in state G on the cell $C_{1,2}$ and $C_{2,1}$ at time $t = 1$. The general in state G initiates a synchronization for the following cells $\{C_{1,2}, C_{1,3}, \dots, C_{1,n}\}$ and $\{C_{2,1}, C_{3,1}, \dots, C_{n,1}\}$, each of length $n - 1$. Note that the length of the array where optimum-time synchronization operations are embedded is shorter by one than the usual embedding in section 2. The cells on the segments are constructed to operate so that they simulate the Mazoyer’s optimum-time synchronization operations. All cells on the two

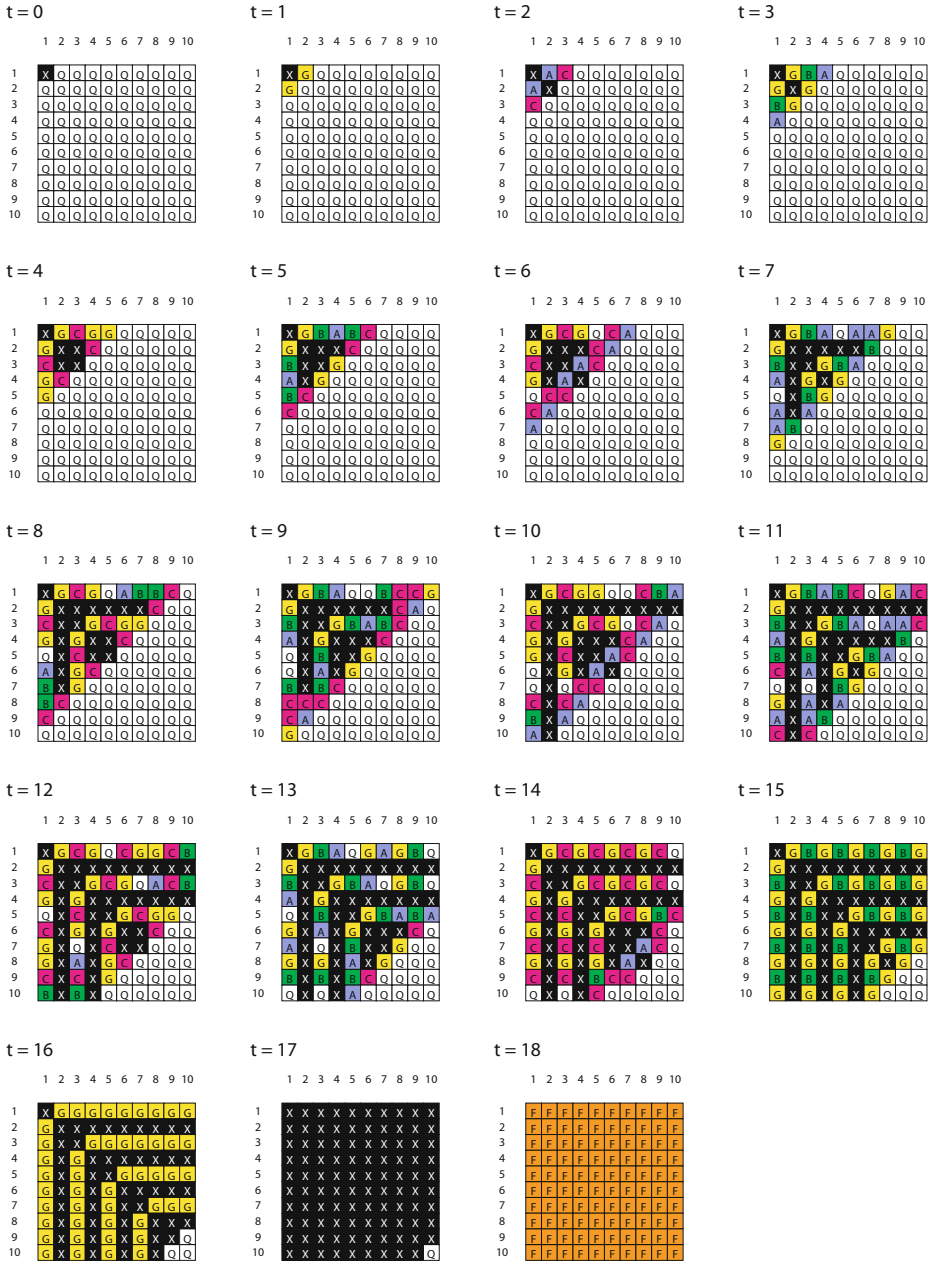


Fig. 5. Snapshots of the synchronization process on 10×10 array

horizontal and vertical segments of length $n - 1$ enter the pre-firing state X at time $t = 1 + 2(n - 1) - 2 = 2n - 3$. In this way, the first L_1 acts as a synchronization layer. At time $t = 2$, the cell $C_{2,2}$ takes the state X and it extends an X -arm (a cell segment in state X) in the right and lower direction, respectively, towards the cells $\{C_{2,3}, C_{2,4}, \dots, C_{2,n}\}$ and $\{C_{3,2}, C_{4,2}, \dots, C_{n,2}\}$, respectively, each of length $n - 2$. Every cell once entered in state X remains unchanged by the time before it meets a local condition for the synchronization given later. At time $t = 2 + n - 2 = n$, the filled-in operation with the stationary state X on the second layer is finished. In this way, the second L_2 acts as a stationary layer.

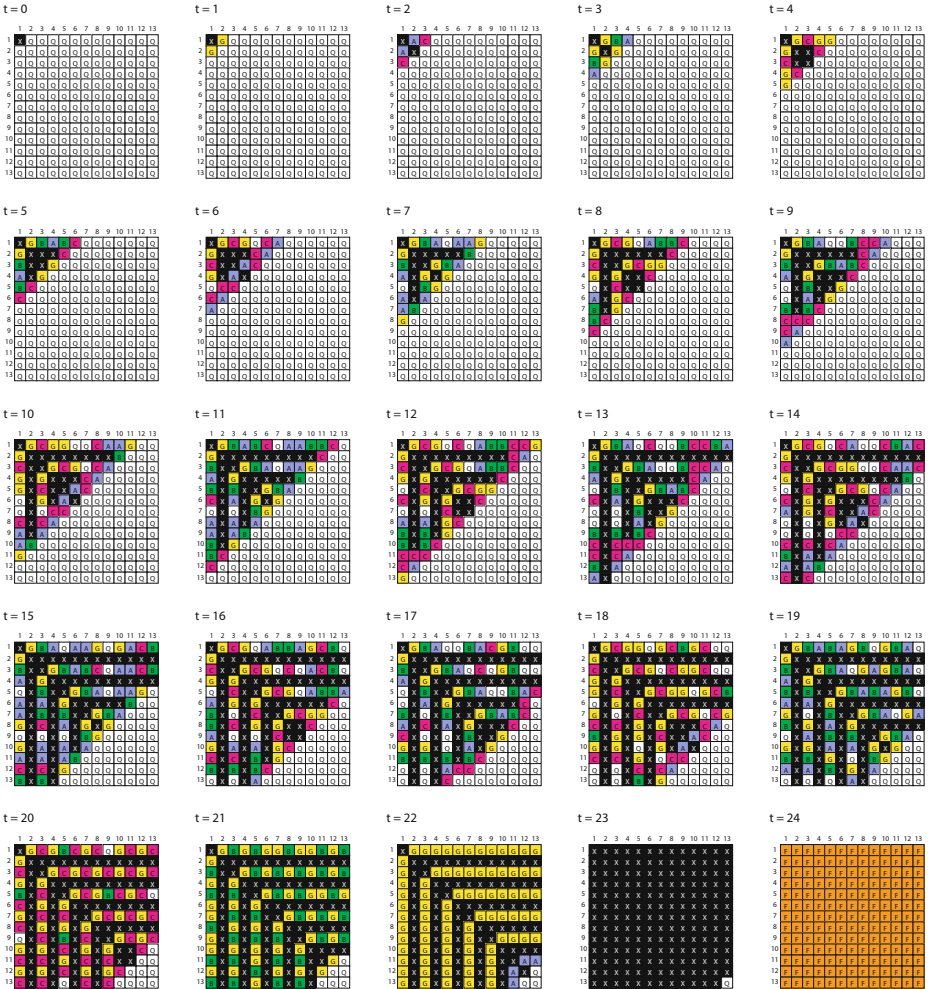


Fig. 6. Snapshots of the synchronization process on 13×13 array

Figure 4 illustrates how the 6-state synchronization and stationary configurations are embedded on a square array of size 7×7 . Concerning the embedding on the odd i th layer, the cell $C_{i,i}$ takes the stationary state X time $t = 2i - 2$ and generates a new general in state G on the cell $C_{i,i+1}$ and $C_{i+1,i}$ at time $t = 2i - 1$. The general in state G initiates a synchronization for the following cells $\{C_{i,i+1}, C_{i,i+2}, \dots, C_{i,n}\}$ and $\{C_{i+1,i}, C_{i+2,i}, \dots, C_{n,i}\}$, each of length $n - i$. All cells on the two horizontal and vertical segments of length $n - i$ enter the pre-firing state X at time $t = 2i - 1 + 2(n - i) - 2 = 2n - 3$. In this way, for odd i , the i th L_i acts as a synchronization layer. As for the even i th layer, at time $t = 2i - 2$, the cell $C_{i,i}$ takes the state X and it extends the X -arm in the right and lower direction, respectively, towards the cells $\{C_{i,i+1}, C_{i,i+2}, \dots, C_{i,n}\}$ and $\{C_{i+1,i}, C_{i+2,i}, \dots, C_{n,i}\}$, each of length $n - i$. Every cell once entered in state X remains unchanged by the time before synchronization. At time $t = 2i - 2 + n - i = n + i - 2$, the filled-in operation on the i th layer for even i is finished.

At time $t = 2n - 3$, all of the cells, except $C_{n,n}$, on the square of size $n \times n$ enter the state X , which is a pre-firing state. The following twelve transition rules, shown in Table 1, are the only ones that falls into the synchronization state F in the last stage. In each 6-tuple rule such that $Y1 \ Y2 \ Y3 \ Y4 \ Y5 \rightarrow Y6$, the symbol $Y1$ denotes the present state of a cell, $Y2$ the east state, $Y3$ the north state, $Y4$ the west state, $Y5$ the south state, and $Y6$ the next state of the cell, respectively. A symbol "*" denotes a boundary state of square arrays.

Table 1. Transition rule set used at the last synchronization step

$Q * X X * \rightarrow F$; $X Q X X * \rightarrow F$; $X X X X X \rightarrow F$; $X X X X * \rightarrow F$;
$X X X * X \rightarrow F$; $X X X * * \rightarrow F$; $X X * X X \rightarrow F$; $X X * * X \rightarrow F$;
$X * X X Q \rightarrow F$; $X * X X X \rightarrow F$; $X * * X Q \rightarrow F$; $X * * X X \rightarrow F$.

Thus we have:

Theorem 4. The seven-state synchronization algorithm \mathcal{A} can synchronize any $n \times n$ square array in optimum $2n - 2$ steps.

Figures 5 and 6 show some snapshots of the synchronization process operating in optimum-steps on 10×10 and 13×13 square arrays. The constructed seven-state cellular automaton has 787 transition rules shown in Appendix I.

4 Conclusions

We have proposed a seven-state optimum-time synchronization algorithm that can synchronize any square arrays of size $n \times n$ with a general at one corner in $2n - 2$ steps. The algorithm is based on a new, simple zebra mapping scheme which embeds one-dimensional synchronization operations onto square arrays.

The seven-state implementation described in terms of state transition table is a smallest realization of time-optimum square synchronizer known at present. The embedding scheme developed in this paper would be useful for state-efficient implementation of multi-dimensional synchronization algorithms, including rectangles and cubic arrays.

Acknowledgments. A part of this work has been supported by Kayamori Foundation of Informational Science Advancement.

References

1. Balzer, R.: An 8-state minimal time solution to the firing squad synchronization problem. *Information and Control* 10, 22–42 (1967)
2. Beyer, W.T.: Recognition of topological invariants by iterative arrays. Ph.D. Thesis, p. 144. MIT (1969)
3. Goto, E.: A minimal time solution of the firing squad problem. Dittoed course notes for Applied Mathematics, vol. 298, pp. 52–59. Harvard University, Cambridge (1962)
4. Mazoyer, J.: A six-state minimal time solution to the firing squad synchronization problem. *Theoretical Computer Science* 50, 183–238 (1987)
5. Moore, E.F.: The firing squad synchronization problem. In: Moore, E.F. (ed.) *Sequential Machines, Selected Papers*, pp. 213–214. Addison-Wesley, Reading (1964)
6. Schmid, H.: Synchronisationsprobleme für zelluläre Automaten mit mehreren Generälen. Diplomarbeit, Universität Karlsruhe (2003)
7. Shinahr, I.: Two- and three-dimensional firing squad synchronization problems. *Information and Control* 24, 163–180 (1974)
8. Szwedinski, H.: Time-optimum solution of the firing-squad-synchronization-problem for n -dimensional rectangles with the general at an arbitrary position. *Theoretical Computer Science* 19, 305–320 (1982)
9. Umeo, H.: Firing squad synchronization algorithms for two-dimensional cellular automata. *Journal of Cellular Automata* 4, 1–20 (2008)
10. Umeo, H.: Firing squad synchronization problem in cellular automata. In: Meyers, R.A. (ed.) *Encyclopedia of Complexity and System Science*, vol. 4, pp. 3537–3574. Springer, Heidelberg (2009)
11. Umeo, H., Hisaoka, M., Akiguchi, S.: Twelve-state optimum-time synchronization algorithm for two-dimensional rectangular cellular arrays. In: Calude, C.S., Dinneen, M.J., Păun, G., Jesús Pérez-Jimenez, M., Rozenberg, G. (eds.) *UC 2005. LNCS*, vol. 3699, pp. 214–223. Springer, Heidelberg (2005)
12. Umeo, H., Maeda, M., Fujiwara, N.: An efficient mapping scheme for embedding any one-dimensional firing squad synchronization algorithm onto two-dimensional arrays. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) *ACRI 2002. LNCS*, vol. 2493, pp. 69–81. Springer, Heidelberg (2002)
13. Umeo, H., Maeda, M., Hisaoka, M., Teraoka, M.: A state-efficient mapping scheme for designing two-dimensional firing squad synchronization algorithms. *Fundamenta Informaticae* 74(4), 603–623 (2006)
14. Umeo, H., Uchino, H.: A new time-optimum synchronization algorithm for rectangle arrays. *Fundamenta Informaticae* 87(2), 155–164 (2008)
15. Umeo, H., Yamawaki, T., Shimizu, N., Uchino, H.: Modeling and simulation of global synchronization processes for large-scale-of two-dimensional cellular arrays. In: *Proc. of Intern. Conf. on Modeling and Simulation, AMS 2007*, pp. 139–144 (2007)

Appendix I: Transition rule set for the 7-state square synchronizer. In each 6-tuple rule such that Y1 Y2 Y3 Y4 Y5 -> Y6, the symbol Y1 denotes the present state, Y2 the east state, Y3 the north state, Y4 the west state, Y5 the south state, and Y6 the next state, respectively. A symbol "*" denotes a boundary state of square arrays (to be continued).

Table with three columns of transition rules. Each rule is a 6-tuple (Y1 Y2 Y3 Y4 Y5 -> Y6) with a state label (Q State, A State, B State) and a transition symbol (Q, X, G, C, B, F). The table lists 1010 such rules.

Appendix I: Transition rule set for the 7-state square synchronizer (continued).

405: B * X G X → G	505: C X A * * → B	605: G G X X X → X	705: X B X G * → X
406: B * * C X → Q	506: C X B X C → C	606: G G * B X → G	706: X C A X G → X
407: B * * G X → Q	507: C X B * * → G	607: G G * C X → A	707: X C B X G → X
	508: C X B * * → G	608: G G * C X → A	708: X C B X G → X
	509: C X B * * → G	609: G G * G X → X	709: X C C X X → X
	510: C X B * * → C	610: G G * X X → X	710: X C G X A → X
	511: C X B * * → C	611: G X Q X A → G	711: X C X Q X → X
	512: C X B * * → G	612: G X Q X B → G	712: X C X Q X → X
	513: C X B * * → G	613: G X Q X C → G	713: X C X A X → X
	514: C X C X Q → C	614: G X Q * A → G	714: X C X B X → X
	515: C X C X A → C	615: G X Q * B → G	715: X C X C X → X
	516: C X C X B → B	616: G X Q * C → G	716: X C X C * → X
	517: C X C X C → C	617: G X A X Q → B	717: X C X G X → X
	518: C X C X * → B	618: G X X B * → G	718: X C G G X → X
	519: C X C * Q → C	619: G X A X C → G	719: X G X Q X → X
	520: C X C * A → A	620: G X A * Q → B	720: X G X A C → X
	521: C X C * B → B	621: G X A * Q → G	721: X G X A X → X
	522: C X C * B → B	622: G X A * Q → G	722: X G X A X → X
	523: C X C * G → B	623: G X B X Q → B	723: X G X B X → X
	524: C X G X Q → B	624: G X B X B → G	724: X G X B * → X
	525: C X G X B → B	625: G X B X C → G	725: X G X C X → X
	526: C X G X G → B	626: G X B B Q → G	726: X G X G X → X
	527: C X G X * → B	627: G X B X * → G	727: X G X G * → X
	528: C X G * Q → B	628: G X B * Q → B	728: X G X X G → X
	529: C X G * G → B	629: G X B * B → G	729: X G * * G → X
	530: C X G * G → B	630: G X B * C → G	730: X X Q X Q → X
	531: C X G * X → B	631: G X X * Q → B	731: X X X A A → X
	532: C X X A Q → B	632: G X B * * → G	732: X X Q X B → X
	533: C * X A X → B	633: G X C X Q → A	733: X X Q X C → X
	534: C * X B X → G	634: G X C X B → G	734: X X Q X Q → X
	535: C * X B X → G	635: G X C X C → G	735: X X X A X → X
	536: C * X G X → G	636: G X C * Q → B	736: X X A A A → X
	537: C * X G X → B	637: G X C X * → A	737: X X A X B → X
	538: C * A X Q → B	638: G X C * Q → A	738: X X A X C → X
	539: C * A X X → B	639: G X C * B → G	739: X X A X G → X
	540: C * B Q X → G	640: G X C * G → A	740: X X X C A → X
	541: C * * B X → G	641: G X C * * → B	741: X X B X A → X
	542: C * * G Q → B	642: G X C * * → A	742: X X B X B → X
	543: C * * G X → B	643: G X G X Q → B	743: X X B X C → X
		644: G X G X B → G	744: X X B X G → X
		645: G X G X C → G	745: X X B X A → X
		646: G X G X G → X	746: X X C X Q → X
		647: G X G X * → X	747: X X C X A → X
		648: G X G * Q → B	748: X X C X B → X
		649: G X G * B → G	749: X X C X C → X
		650: G X G * C → G	750: X X C X G → X
		651: G X G * G → X	751: X X C X X → X
		652: G X G * * → X	752: X X C G X → X
		653: G X X X B → G	753: X X X G X → X
		654: G X X X C → B	754: X X G X A → X
		655: G X X X G → X	755: X X G X B → X
		656: G X X * B → G	756: X X G X C → X
		657: G X X * C → G	757: X X G X G → X
		658: G X X * G → X	758: X X G X X → X
		659: G * X B Q → G	759: X X X B X → X
		660: G * X B X → G	760: X X X C X → X
		661: G * X C X → A	761: X X X C C → X
		662: G * X C Q → A	762: X X X G X → X
		663: G * X G A → X	763: X X X X X → F
		664: G * * B Q → G	764: X X X X * → F
		665: G * * B X → G	765: X X X X * → F
		666: G * * C X → A	766: X X X X * → F
		667: G * * C Q → A	767: X X * X X → F
		668: G * * G X → X	768: X X * * X → F
			769: X * X X Q → X
			770: X * Q X A → X
			771: X * Q X B → X
			772: X * * Q C → X
			773: X * A X Q → X
			774: X * * A X → X
			775: X * * B X → X
			776: X * B X B → X
			777: X * * B X → G
			778: X * * C X → X
			779: X * * C X → X
			780: X * G X Q → X
			781: * * G X A → X
			782: X * * G X → B
			783: X * * G X → G
			784: X * * X X → F
			785: X * X X X → F
			786: X * * X Q → F
			787: X * * X X → F

Null Boundary 90/150 Cellular Automata for Multi-byte Error Correcting Code

Jaydeb Bhaumik, Dipanwita Roy Chowdhury, and Indrajit Chakrabarti

Indian Institute of Technology, Kharagpur-721302, India

Abstract. Cellular Automata(CA) is a well known tool to generate byte error correcting code. In this paper, we propose a CA-based multi-byte Error Correcting Code (ECC) which overcomes the weaknesses and limitation of existing scheme. As a case study three and four bytes ECC are discussed in detailed. A complete decoding algorithm of CA-based 3-byte error correcting code is presented in this work. Proposed 3-byte ECC scheme can correct errors when errors are distributed within information and check bytes or concentrated in any one of them. In case of CA-based 4-byte ECC, at most 4-byte errors can be corrected if all the errors are concentrated in information or check bytes.

1 Introduction

Error correcting codes have a wide range of applications in digital data communications, memory system design [1], fault tolerant computer design etc. Reed-Solomon (RS) code is a well known non-binary, block code and popularly used for error correction in many applications like wireless communications, high speed modems and storage devices (CD, DVD). A number of general encoding and decoding schemes of the RS codes may be found in the literature [10]. Many researchers have put their effort to minimize the complexity of RS decoder for communication applications.

Cellular automata has already established its novelty for bits and bytes error correcting codes [2][3]. A scheme for pipeline implementation of CA-based t -byte error correcting and t -byte error detecting codes has been proposed in [4]. Another scheme for parallel implementation of CA based single byte error correcting-double byte error detecting, double byte error correcting-double byte error detecting code has been reported in [5]. Application of $GF(2^p)$ CA in burst error correcting codes has been reported in [6]. The CA-based byte error correcting code in [3] is simpler to design compared to other schemes. However, a few mistakes have been identified in the error location and error magnitude computation algorithm of scheme [3]. In scheme [3] for $t = 2, 3, 4$, single equation between syndromes is used to compute the error locations which gives a set of solutions instead of a single solution. Also the scheme [3] has one limitation that decoder can correct t -byte errors ($t \geq 2$) provided errors are totally confined to information or check bytes only. An improved scheme has been proposed in [7], which eliminates the weaknesses and limitation of the previous scheme [3], for 2-byte error detection and correction only. VLSI implementation of CA-based

improved double byte error correcting encoder and decoder [7] is presented in [8]. It is mentioned that the scheme for CA-based 2-byte ECC can be extended to 3-byte ECC, however the proposal [8] is for a restricted situation. The restriction is that the determination of error locations and the computation of error values are possible only if all the errors are concentrated in information bytes only. It is also mentioned that errors can be corrected if errors are distributed between information and check bytes. But the full decoding algorithm for 3-byte ECC has not been provided in [8].

In this paper, we propose a multi-byte ECC using CA and as a case study, three and four bytes ECC are discussed in detailed. A full decoding algorithm for a 3-byte ECC is proposed, which can detect and correct at most three errors and the scheme is independent of error position i.e. whether errors are distributed between information and check bytes or concentrated in information bytes only. Also the scheme has been extended for 4-byte ECC. Maximum length CA is essential for CA-based byte error correcting code. For an 8-bit maximum length CA the characteristic polynomial is a primitive polynomial of degree 8. There exist 16 primitive polynomials in $GF(2^8)$, with the coefficients of the polynomials are in $GF(2)$. Different codes can be generated using different primitive polynomials. These set of codes are required in several cryptographic applications for better security. One such example is presented in [9], where an integrated code is used for both error correction and message authentication. In this paper, we have computed one such maximum length CA rule vector for each primitive polynomial in $GF(2^8)$ based on CA-rules 90 and 150. These 8-bit CA rule vectors can be considered as a toy example for the scheme [9].

The rest of this paper is organized as follows. In the next section, proposed CA based multi-byte error correcting code is described. The paper is concluded in section 3.

2 CA-Based Multi-byte Error Correcting Code

In case of t -byte ECC in $GF(2^8)$, it is essential to send $2t$ number of check bytes with the block of information bytes. Each check byte C_b is computed by running an 8-bit CA with characteristic matrix T^b for N cycles while sequentially feeding N information bytes using the expression.

$$C_b = T^b(D_{N-1} \oplus T^b(D_{N-2} \oplus \dots \oplus T^b(D_1 \oplus T^b(0 \oplus D_0)) \dots)) \quad (1)$$

where $0 \leq b \leq (2t - 1)$ and T is the characteristic matrix of an 8 cell maximum length CA. The comp-check-byte algorithm explains method for computing C_b . Decoder computes the b -th syndrome byte S_b using the following equation.

$$S_b = C_b \oplus C'_b; \quad 0 \leq b \leq (2t - 1). \quad (2)$$

where C_b is the b -th received check byte and C'_b is the b -th check byte recomputed from the received information bytes. Assume D'_m , E_m are the received m -th information byte and the calculated m -th error byte respectively then the corrected information byte is obtained by using the equation as follows.

$$D_m = D'_m \oplus E_m; \quad \text{where } 0 \leq m \leq (N - 1) \tag{3}$$

Comp-check-byte: check byte C_b computation algorithm

s denotes the state of the 8 bits CA

begin

$s := 0$; **for** $k = 0$ *to* $N - 1$ **do**

begin

$s := s \oplus D_k$;

Run the CA for one cycle; (CA with characteristic matrix T^b)

end;

$C_b := s$;

end;

2.1 8-bit CA Rule Vectors for All Primitive Polynomials

Proposed CA-based byte error correcting code is based on null boundary maximum length CA. Therefore, rule vectors for the 8-bit maximum length null boundary CA is discussed in this section. We have considered the CA-rules 90 and 150 only. By simulation, we have found the rule vectors for each primitive polynomial in $GF(2^8)$ and they have been listed in Table 1. In Table 1, ‘0’ and ‘1’ correspond to rule 90 and 150 respectively. It has been observed that the mirror image of each rule vector corresponds to the same primitive polynomial. For example, 00000110 and 01100000 are two rule vectors for primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$, where the two rule vectors are mirror image of each other. Different rule vectors generate different codewords for the same data block and there is an option to select any one of the 16 rule vectors given in Table 1. As a case study, CA-based 3-byte and 4-byte error correcting codes are discussed in following subsection.

2.2 3-Byte Error Correcting Code

This section explains the complete decoding algorithm for the CA-based three bytes error correcting code. Proposed scheme can detect and correct at most three errors and it is independent of error positions. In three byte error correcting code, six check bytes are generated by using (1) and the syndrome values are computed using (2). Depending on the number of syndromes having value non-zero and zero, some conclusions may be drawn. The results are shown in Table 2. In case of 3-byte ECC, four possibilities may occur: no error, one error, two errors and three errors. The obvious distribution of errors are summarized in Table 3 to make the analysis more systematic. In the case of three errors in three check bytes is identified when three syndromes have non-zero value and other three have zero value. The result is also shown in Table 2. All syndromes having value zero indicates there is no error. In this section, we first describe the decoding method where all three errors are concentrated within information bytes, from [8] for the sake of completeness.

Table 1. 8-bit CA-rule vectors for all primitive polynomials in $GF(2^8)$

No.	Primitive Polynomial	Corresponding CA-rule vector
1	$x^8 + x^4 + x^3 + x^2 + 1$	00000110
2	$x^8 + x^5 + x^3 + x + 1$	01011111
3	$x^8 + x^5 + x^3 + x^2 + 1$	01110111
4	$x^8 + x^6 + x^3 + x^2 + 1$	01101100
5	$x^8 + x^6 + x^4 + x^3 + x^2 + x + 1$	10010011
6	$x^8 + x^6 + x^5 + x + 1$	11010010
7	$x^8 + x^6 + x^5 + x^2 + 1$	00101101
8	$x^8 + x^6 + x^5 + x^3 + 1$	00001111
9	$x^8 + x^6 + x^5 + x^4 + 1$	00111001
10	$x^8 + x^7 + x^2 + x + 1$	11101111
11	$x^8 + x^7 + x^3 + x^2 + 1$	00101010
12	$x^8 + x^7 + x^5 + x^3 + 1$	01000101
13	$x^8 + x^7 + x^6 + x + 1$	01011101
14	$x^8 + x^7 + x^6 + x^3 + x^2 + x + 1$	01011011
15	$x^8 + x^7 + x^6 + x^5 + x^2 + x + 1$	11001011
16	$x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + 1$	11010101

Table 2. Number of non-zero syndromes *vs.* decision

Number of syndromes		Decision
Value zero	Value non-zero	
6	0	no error
5	1	1 check byte error
4	2	2 check bytes error
3	3	3 check bytes error
2	4	4 check bytes error / 2 check bytes and 1 information byte
1	5	two/three bytes error
0	6	one/two/three bytes error

Table 3. Error distribution

Total number of errors in bytes	Number of errors in	
	information bytes	check bytes
3	3	0
	0	3
	2	1
	1	2
2	2	0
	0	2
	1	1
1	1	0
	0	1

Three information bytes error. Suppose E_m, E_n and E_o are the error magnitudes in the m -th, n -th, and o -th information bytes respectively with $m \neq n \neq o$. Then the corresponding syndrome equations are as follows

$$\begin{aligned} S_0 &= E_m \oplus E_n \oplus E_o ; S_1 = T^i E_m \oplus T^j E_n \oplus T^k E_o \\ S_2 &= T^{2i} E_m \oplus T^{2j} E_n \oplus T^{2k} E_o ; S_3 = T^{3i} E_m \oplus T^{3j} E_n \oplus T^{3k} E_o \\ S_4 &= T^{4i} E_m \oplus T^{4j} E_n \oplus T^{4k} E_o ; S_5 = T^{5i} E_m \oplus T^{5j} E_n \oplus T^{5k} E_o \end{aligned} \quad (4)$$

where S_0, S_1, S_2, S_3, S_4 and S_5 are six syndrome bytes and $i + m = N, j + n = N, k + o = N$ and N is the number of information bytes in the codeword. The required equations to compute the error locations are as follows.

$$\begin{aligned} T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5 &= T^{2k}(T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \\ T^{2j}(T^{2i} S_1 \oplus S_3) \oplus T^{2i} S_3 \oplus S_5 &= T^k(T^{2j}(T^{2i} S_0 \oplus S_2) \oplus T^{2i} S_2 \oplus S_4) \\ T^j(T^{3i} S_1 \oplus S_4) \oplus T^{3i} S_2 \oplus S_5 &= T^k(T^j(T^{3i} S_0 \oplus S_3) \oplus T^{3i} S_1 \oplus S_4) \end{aligned} \quad (5)$$

Simultaneous solution of three equations estimates the three error locations. Error magnitudes in three bytes m, n and o are calculated using the following equations.

$$\begin{aligned} E_n &= T^{L-a}(T^{2k}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \\ E_o &= T^{L-b}(T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3); E_m = S_0 \oplus E_o \oplus E_n \end{aligned} \quad (6)$$

where $T^a = (T^i \oplus T^j)(T^{2j} \oplus T^{2k})$, $T^b = (T^i \oplus T^k)(T^{2j} \oplus T^{2k})$ and $L = 2^8 - 1$ is the cycle length of an 8-cell maximum length group CA.

Two information bytes and one check byte error. There are six possibilities when any one of the six check bytes and any two information bytes are erroneous. Assume e_0, E_m and E_n are the errors in check byte C_0, m -th and n -th information bytes respectively. Then we can write the syndrome equations as follows.

$$\begin{aligned} S_0 &= E_m \oplus E_n \oplus e_0 ; S_1 = T^i E_m \oplus T^j E_n ; S_2 = T^{2i} E_m \oplus T^{2j} E_n \\ S_3 &= T^{3i} E_m \oplus T^{3j} E_n ; S_4 = T^{4i} E_m \oplus T^{4j} E_n ; S_5 = T^{5i} E_m \oplus T^{5j} E_n \end{aligned} \quad (7)$$

From the Equations in (7), we get

$$\begin{aligned} T^i S_0 \oplus S_1 &= (T^i \oplus T^j) E_n \oplus T^i e_0 ; T^i S_2 \oplus S_3 = T^{2j}(T^i \oplus T^j) E_n \\ T^i S_4 \oplus S_5 &= T^{4j}(T^i \oplus T^j) E_n ; T^{2i} S_0 \oplus S_2 = (T^{2i} \oplus T^{2j}) E_n \oplus T^{2i} e_0 \\ T^{2i} S_1 \oplus S_3 &= T^j(T^{2i} \oplus T^{2j}) E_n ; T^{2i} S_2 \oplus S_4 = T^{2j}(T^{2i} \oplus T^{2j}) E_n \\ T^{2i} S_3 \oplus S_5 &= T^{3j}(T^{2i} \oplus T^{2j}) E_n ; T^{3i} S_0 \oplus S_3 = (T^{3i} \oplus T^{3j}) E_n \oplus T^{3i} e_0 \\ T^{3i} S_1 \oplus S_4 &= T^j(T^{3i} \oplus T^{3j}) E_n ; T^{3i} S_2 \oplus S_5 = T^{2j}(T^{3i} \oplus T^{3j}) E_n \end{aligned} \quad (8)$$

Combining the equations in (8), we get

$$\begin{aligned} T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5 &= 0; T^{2j}(T^{2i} S_0 \oplus S_2) \oplus T^{2i} S_2 \oplus S_4 = T^{2j}(T^{2i} e_0) \\ T^j(T^{3i} S_0 \oplus S_3) \oplus T^{3i} S_1 \oplus S_4 &= T^j(T^{3i} e_0); T^{2j}(T^{2i} S_1 \oplus S_3) \oplus T^{2i} S_3 \oplus S_5 = 0 \\ T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3 &= T^{2j}(T^i e_0); T^j(T^{3i} S_1 \oplus S_4) \oplus T^{3i} S_2 \oplus S_5 = 0 \end{aligned} \quad (9)$$

Table 4. Equations for computing at most two errors in information bytes

U	V	W	X	Y	Z	Errors in	Error value
0	1	1	0	1	0	C_0, D_m, D_n	$E_n = T^{(L-\alpha)}(T^i S_1 \oplus S_2), E_m = T^{(L-\beta)}(T^j S_1 \oplus S_2)$
0	0	1	1	1	1	C_1, D_m, D_n	$E_n = T^{(L-\gamma)}(T^{2i} S_0 \oplus S_2), E_m = S_0 \oplus E_n$
1	1	0	0	1	1	C_2, D_m, D_n	$E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1), E_m = S_0 \oplus E_n$
1	0	1	1	1	0	C_3, D_m, D_n	$E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1), E_m = S_0 \oplus E_n$
1	1	1	0	0	1	C_4, D_m, D_n	$E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1), E_m = S_0 \oplus E_n$
1	0	0	1	0	1	C_5, D_m, D_n	$E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1), E_m = S_0 \oplus E_n$
0	0	0	0	0	0	D_m, D_n	$E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1), E_m = S_0 \oplus E_n$

$$\begin{aligned}
 U &= T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5, V = T^{2j}(T^{2i} S_0 \oplus S_2) \oplus T^{2i} S_2 \oplus S_4 \\
 W &= T^j(T^{3i} S_0 \oplus S_3) \oplus T^{3i} S_1 \oplus S_4, X = T^{2j}(T^{2i} S_1 \oplus S_3) \oplus T^{2i} S_3 \oplus S_5 \\
 Y &= T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3, Z = T^j(T^{3i} S_1 \oplus S_4) \oplus T^{3i} S_2 \oplus S_5 \\
 T^\alpha &= T^j(T^i \oplus T^j), T^\beta = T^i(T^i \oplus T^j), T^\gamma = (T^{2i} \oplus T^{2j}), T^\delta = (T^i \oplus T^j)
 \end{aligned}$$

Equations in (9) are used to determine the error locations $m = N - i$ and $n = N - j$. From the syndrome equations in (7), we get

$$\begin{aligned}
 T^i S_1 \oplus S_2 &= T^j(T^i \oplus T^j)E_n \text{ or } E_n = T^{L-\alpha}(T^i S_1 \oplus S_2) \\
 T^j S_1 \oplus S_2 &= T^i(T^i \oplus T^j)E_m \text{ or } E_m = T^{L-\beta}(T^j S_1 \oplus S_2)
 \end{aligned} \tag{10}$$

where $T^\alpha = T^j(T^i \oplus T^j)$, $T^\beta = T^i(T^i \oplus T^j)$ and $L = 2^8 - 1$. Error magnitudes E_m and E_n are computed using equations in (10), Equations to determine the error locations and the values for the other five cases and the case discussed in this section are summarized in Table 4. Zero and non-zero value are represented by ‘0’ and ‘1’ in Table 4. For any $e_b \neq 0$, $T^p e_b \neq 0$, where $1 \leq p \leq 2^8 - 1$ and e_b is the error in b -th check byte. Hence $V = 1, W = 1$ and $Y = 1$ in the first row of Table 4.

Two information bytes error. Assume E_m and E_n are the errors in the m -th and n -th information byte respectively. Then the syndromes are as follows.

$$\begin{aligned}
 S_0 &= E_m \oplus E_n ; S_1 = T^i E_m \oplus T^j E_n ; S_2 = T^{2i} E_m \oplus T^{2j} E_n \\
 S_3 &= T^{3i} E_m \oplus T^{3j} E_n ; S_4 = T^{4i} E_m \oplus T^{4j} E_n ; S_5 = T^{5i} E_m \oplus T^{5j} E_n
 \end{aligned} \tag{11}$$

Combining the equations in (11), we get

$$\begin{aligned}
 T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5 &= 0; T^{2j}(T^{2i} S_0 \oplus S_2) \oplus T^{2i} S_2 \oplus S_4 = 0 \\
 T^j(T^{3i} S_0 \oplus S_3) \oplus T^{3i} S_1 \oplus S_4 &= 0; T^{2j}(T^{2i} S_1 \oplus S_3) \oplus T^{2i} S_3 \oplus S_5 = 0 \\
 T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3 &= 0; T^j(T^{3i} S_1 \oplus S_4) \oplus T^{3i} S_2 \oplus S_5 = 0
 \end{aligned} \tag{12}$$

The error locations $m = N - i$ and $n = N - j$ are determined using the equations in (12). Using the syndrome equations in (11), we can write

$$T^i S_0 \oplus S_1 = (T^i \oplus T^j)E_n \text{ or } E_n = T^{(L-\delta)}(T^i S_0 \oplus S_1); E_m = S_0 \oplus E_n \tag{13}$$

where $T^\delta = T^i \oplus T^j$ and $L = 2^8 - 1$. Error magnitudes may be found using equation in (I3). Equations are also shown in the last row of Table 4.

One information byte and two check bytes error. There are fifteen situations when any two check bytes and any one information byte are erroneous. In this section, out of the fifteen cases, the proof of one case is given as follows and the results for the other cases are given in Table 5 for the sake of brevity. Assume e_0, e_1 and E_m are the error values in the check bytes C_0, C_1 and the m -th information byte respectively. Hence, the corresponding syndrome equations are as follows.

$$\begin{aligned} S_0 &= E_m \oplus e_0 ; S_1 = T^i E_m \oplus e_1 ; S_2 = T^{2i} E_m \\ S_3 &= T^{3i} E_m ; S_4 = T^{4i} E_m ; S_5 = T^{5i} E_m \end{aligned} \tag{14}$$

From the syndrome equations in (I4), we get

$$\begin{aligned} T^i S_0 \oplus S_1 &= T^i e_0 \oplus e_1 ; T^i S_2 \oplus S_3 = 0 ; T^i S_4 \oplus S_5 = 0 ; T^{2i} S_2 \oplus S_4 = 0 \\ T^{2i} S_3 \oplus S_5 &= 0 ; T^{2i} S_0 \oplus S_2 = T^{2i} e_0 ; T^{2i} S_1 \oplus S_3 = T^{2i} e_1 \\ T^{3i} S_1 \oplus S_4 &= T^{3i} e_1 ; T^{3i} S_2 \oplus S_5 = 0 ; T^{3i} S_0 \oplus S_3 = T^{3i} e_0 \end{aligned} \tag{15}$$

For any non-zero value of e_0 and e_1 , $T^g e_0 \neq 0$ and $T^h e_1 \neq 0$, where $1 \leq g, h \leq (2^8 - 1)$. Let $A = T^i e_0 \oplus e_1$; then the value of A may or may not be zero, because it depends on the value of e_0, e_1 and i . In Table 5, x, 1 and 0 indicate the don't care, non-zero value and zero value respectively.

One information byte error. Consider the case when only one information byte is erroneous. Syndrome equations in this case are as follows.

$$\begin{aligned} S_0 &= E_m ; S_1 = T^i E_m ; S_2 = T^{2i} E_m \\ S_3 &= T^{3i} E_m ; S_4 = T^{4i} E_m ; S_5 = T^{5i} E_m \end{aligned} \tag{16}$$

Combining the syndrome equations in (I6), we get

$$\begin{aligned} T^i S_0 \oplus S_1 &= 0 ; T^i S_2 \oplus S_3 = 0 ; T^i S_4 \oplus S_5 = 0 ; T^{2i} S_2 \oplus S_4 = 0 \\ T^{2i} S_3 \oplus S_5 &= 0 ; T^{2i} S_0 \oplus S_2 = 0 ; T^{2i} S_1 \oplus S_3 = 0 ; T^{3i} S_1 \oplus S_4 = 0 \\ T^{3i} S_2 \oplus S_5 &= 0 ; T^{3i} S_0 \oplus S_3 = 0 \end{aligned} \tag{17}$$

If all the equations in (I7) are satisfied, then one error in $(N - i)$ -th information byte is identified and the corresponding error magnitude is $E_m = S_0$.

One information byte and one check byte error. There are six situations in which any one of the information bytes and any one of the check bytes are erroneous. Consider the case when E_m and e_0 are the errors in m -th information byte and the check byte C_0 respectively. Syndromes are computed using (II).

$$\begin{aligned} S_0 &= E_m \oplus e_0 ; S_1 = T^i E_m ; S_2 = T^{2i} E_m \\ S_3 &= T^{3i} E_m ; S_4 = T^{4i} E_m ; S_5 = T^{5i} E_m \end{aligned} \tag{18}$$

Table 5. Equations for computing at most one error in information byte

A	B	C	D	E	F	G	H	I	J	Errors in	Error value
x	0	0	0	0	1	1	1	0	1	C_0, C_1, D_m	$T^{L-2i}S_2$
1	1	0	1	0	x	0	0	1	1	C_0, C_2, D_m	$T^{L-i}S_1$
1	1	0	0	1	1	1	0	0	x	C_0, C_3, D_m	$T^{L-i}S_1$
1	0	1	1	0	1	0	1	0	1	C_0, C_4, D_m	$T^{L-i}S_1$
1	0	1	0	1	1	0	0	1	1	C_0, C_5, D_m	$T^{L-i}S_1$
1	1	0	1	0	1	1	1	1	0	C_1, C_2, D_m	S_0
1	1	0	0	1	0	x	1	0	1	C_1, C_3, D_m	S_0
1	0	1	1	0	0	1	x	0	0	C_1, C_4, D_m	S_0
1	0	1	0	1	0	1	1	1	0	C_1, C_5, D_m	S_0
0	x	0	1	1	1	1	0	1	1	C_2, C_3, D_m	S_0
0	1	1	x	0	1	0	1	1	0	C_2, C_4, D_m	S_0
0	0	1	1	1	1	0	0	x	0	C_2, C_5, D_m	S_0
0	1	1	1	1	0	1	1	0	1	C_3, C_4, D_m	S_0
0	1	1	0	x	0	1	0	1	1	C_3, C_5, D_m	S_0
0	0	x	1	1	0	0	1	1	0	C_4, C_5, D_m	S_0
0	0	0	0	0	0	0	0	0	0	D_m	S_0
1	0	0	0	0	1	0	0	0	1	C_0, D_m	$T^{L-2i}S_1$
1	0	0	0	0	0	1	1	0	0	C_1, D_m	S_0
0	1	0	1	0	1	0	0	1	0	C_2, D_m	S_0
0	1	0	0	1	0	1	0	0	1	C_3, D_m	S_0
0	0	1	1	0	0	0	1	0	0	C_4, D_m	S_0
0	0	1	0	1	0	0	0	1	0	C_5, D_m	S_0

$$\begin{aligned}
 A &= T^i S_0 \oplus S_1, B = T^i S_2 \oplus S_3, C = T^i S_4 \oplus S_5, D = T^{2i} S_2 \oplus S_4 \\
 E &= T^{2i} S_3 \oplus S_5, F = T^{2i} S_0 \oplus S_2, G = T^{2i} S_1 \oplus S_3, H = T^{3i} S_1 \oplus S_4 \\
 I &= T^{3i} S_2 \oplus S_5, \text{ and } J = T^{3i} S_0 \oplus S_3
 \end{aligned}$$

Combining the syndrome equations in (18), we get

$$\begin{aligned}
 T^i S_0 \oplus S_1 &= T^i e_0 \neq 0; T^i S_2 \oplus S_3 = 0; T^i S_4 \oplus S_5 = 0; T^{2i} S_2 \oplus S_4 = 0 \\
 T^{2i} S_3 \oplus S_5 &= 0; T^{2i} S_0 \oplus S_2 = T^{2i} e_0 \neq 0; T^{2i} S_1 \oplus S_3 = 0 \\
 T^{3i} S_1 \oplus S_4 &= 0; T^{3i} S_2 \oplus S_5 = 0; T^{3i} S_0 \oplus S_3 = T^{3i} e_0 \neq 0
 \end{aligned} \tag{19}$$

If all the equations in (19) are satisfied, then one error in $(N - i)$ -th information byte and another error in check byte C_0 are identified. The error magnitude is $E_m = T^{L-i}S_1$. The required equations for the other five cases are summarized in Table 5 for the sake of brevity. Therefore, it is possible to locate and correct all the three-byte errors. It is noted that all expressions A, B, \dots, J in Table 5 and all expressions U, V, \dots, Z in Table 4 are part of the equations in (5). The following section describes the CA-based 4-byte ECC.

2.3 4-Byte Error Correcting Code

In case of CA-based 4-byte error correcting code, eight check bytes are computed using (1) and the decoder computes the eight syndrome bytes employing (2). In this section, we derive the equations to determine the error locations and the error magnitudes, provided the errors are concentrated in the information bytes only. Suppose $S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7$ are the eight syndromes and E_m, E_n, E_o, E_p are the errors in the m -th, n -th, o -th, p -th information bytes respectively, where $m \neq n \neq o \neq p$.

$$\begin{aligned}
 S_0 &= E_m \oplus E_n \oplus E_o \oplus E_p \\
 S_1 &= T^i E_m \oplus T^j E_n \oplus T^k E_o \oplus T^l E_p \\
 S_2 &= T^{2i} E_m \oplus T^{2j} E_n \oplus T^{2k} E_o \oplus T^{2l} E_p \\
 S_3 &= T^{3i} E_m \oplus T^{3j} E_n \oplus T^{3k} E_o \oplus T^{3l} E_p \\
 S_4 &= T^{4i} E_m \oplus T^{4j} E_n \oplus T^{4k} E_o \oplus T^{4l} E_p \\
 S_5 &= T^{5i} E_m \oplus T^{5j} E_n \oplus T^{5k} E_o \oplus T^{5l} E_p \\
 S_6 &= T^{6i} E_m \oplus T^{6j} E_n \oplus T^{6k} E_o \oplus T^{6l} E_p \\
 S_7 &= T^{7i} E_m \oplus T^{7j} E_n \oplus T^{7k} E_o \oplus T^{7l} E_p
 \end{aligned} \tag{20}$$

Using the syndrome equations in (20), the following four equations can be formulated to compute four error locations, provided all the errors are concentrated in the information bytes only.

$$\begin{aligned}
 &T^{2k}(T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5) \oplus T^{2j}(T^i S_4 \oplus S_5) \oplus T^i S_6 \oplus S_7 = \\
 &T^{2l}(T^{2k}(T^{2j}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \oplus T^{2j}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5); \\
 &T^k(T^j(T^{4i} S_1 \oplus S_5) \oplus T^{4i} S_2 \oplus S_6) \oplus T^j(T^{4i} S_2 \oplus S_6) \oplus T^{4i} S_3 \oplus S_7 = \\
 &T^l(T^k(T^j(T^{4i} S_0 \oplus S_4) \oplus T^{4i} S_1 \oplus S_5) \oplus T^j(T^{4i} S_1 \oplus S_5) \oplus T^{4i} S_2 \oplus S_6); \\
 &T^k(T^j(T^{3i} S_2 \oplus S_5) \oplus T^{3i} S_3 \oplus S_6) \oplus T^j(T^{3i} S_3 \oplus S_6) \oplus T^{3i} S_4 \oplus S_7 = \\
 &T^{2l}(T^k(T^j(T^{3i} S_0 \oplus S_3) \oplus T^{3i} S_1 \oplus S_4) \oplus T^j(T^{3i} S_1 \oplus S_4) \oplus T^{3i} S_2 \oplus S_5); \\
 &T^{2k}(T^j(T^{2i} S_2 \oplus S_4) \oplus T^{2i} S_3 \oplus S_5) \oplus T^j(T^{2i} S_4 \oplus S_6) \oplus T^{2i} S_5 \oplus S_7 = \\
 &T^{2l}(T^{2k}(T^j(T^{2i} S_0 \oplus S_2) \oplus T^{2i} S_1 \oplus S_3) \oplus T^j(T^{2i} S_2 \oplus S_4) \oplus T^{2i} S_3 \oplus S_5)
 \end{aligned} \tag{21}$$

Simultaneous solution of the equations in (21) determines the four error positions within the information bytes, where $m = N - i, n = N - j, o = N - k$ and $p = N - l$ and N is the number of information bytes in the codeword. The four error magnitudes E_m, E_n, E_o and E_p are calculated using the following equations.

$$\begin{aligned}
 E_n &= T^{L-a}(T^{2k}(T^{2l}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \oplus T^{2l}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5) \\
 E_o &= T^{L-b}(T^{2j}(T^{2l}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \oplus T^{2l}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5) \\
 E_p &= T^{L-c}(T^{2j}(T^{2k}(T^i S_0 \oplus S_1) \oplus T^i S_2 \oplus S_3) \oplus T^{2k}(T^i S_2 \oplus S_3) \oplus T^i S_4 \oplus S_5) \\
 E_m &= S_0 \oplus E_n \oplus E_o \oplus E_p
 \end{aligned} \tag{22}$$

where $T^a = (T^j \oplus T^i)(T^{2j} \oplus T^{2k})(T^{2j} \oplus T^{2l})$; $T^b = (T^k \oplus T^i)(T^{2j} \oplus T^{2k})(T^{2k} \oplus T^{2l})$; $T^c = (T^l \oplus T^i)(T^{2j} \oplus T^{2l})(T^{2k} \oplus T^{2l})$ and $L = 2^8 - 1$ is the cycle length of

an 8-cell maximum length group CA. Similar to a 3-byte ECC, it is possible to correct the errors when the errors are distributed between information and the check bytes.

One disadvantage of the proposed error correcting code is that the error location identification block has a time complexity of N^t , where N is the number of information bytes and t is the number of errors to be corrected. But the decoding time can be reduced by duplicating the error location identification module.

3 Conclusion

This paper presents an improved scheme for multi-byte error correcting code using CA which overcomes the weaknesses and limitation of existing scheme. As a case study full decoding algorithm for three byte error correcting code is presented. Proposed CA-based four byte error correcting code can detect and correct at most 4 errors if all errors are concentrated in information or check bytes.

References

1. Chen, C.L.: Error-correcting codes for byte-organized memory systems. *IEEE Trans. Information Theory* 32(2), 181–185 (1986)
2. Roy Chowdhury, D., Basu, S., Sen Gupta, I., Chaudhuri, P.P.: Design of CAECC-Cellular Automata Based Error Correcting Code. *IEEE Transaction on Computers* 43(6), 759–764 (1994)
3. Roy Chowdhury, D., Sen Gupta, I., Chaudhuri, P.P.: CA-Based Byte Error-Correcting code. *IEEE Transaction on Computers* 44(3), 371–382 (1995)
4. Nandi, S., Rambabu, C., Chaudhuri, P.P.: A VLSI Architecture for Cellular Automata Based Reed-Solomon Decoder. In: 4th International Symposium on Parallel Architecture, Algorithm and Networks, Australia, pp. 158–165 (1999)
5. Sasidhar, K., Chattopadhyay, S., Chaudhuri, P.P.: CAA Decoder for Cellular Automata Based Byte Error Correcting Code. *IEEE Transaction on Computers* 45(9), 1003–1016 (1996)
6. Paul, K., Roy Chowdhury, D.: Application of $GF(2^p)$ CA in Burst Error Correcting Codes. In: 13th International Conference on VLSI Design, India, pp. 562–567 (2000)
7. Bhaumik, J., Roy Chowdhury, D., Chakrabarti, I.: An Improved Double Byte Error Correcting Code using Cellular Automata. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 463–470. Springer, Heidelberg (2008)
8. Bhaumik, J., Roy Chowdhury, D.: New Architectural Design of CA-Based Codec. *IEEE Transactions on Very Large Scale Integration Systems* 18(7), 1139–1144 (2010)
9. Bhaumik, J., Roy Chowdhury, D.: An Integrated ECC-MAC Based on RS Code. In: Gavrilova, M.L., Tan, C.J.K., Moreno, E.D. (eds.) *Transactions on Computational Science IV*. LNCS, vol. 5430, pp. 117–135. Springer, Heidelberg (2009)
10. Lin, S., Costello, D.J.: *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, Inc., Englewood Cliffs (1983)

Generating Cryptographically Suitable Non-linear Maximum Length Cellular Automata

Sourav Das^{1,2} and Dipanwita Roy Chowdhury¹

¹ Dept. of CSE, Indian Institute of Technology, Kharagpur, India
drc@cse.iitkgp.ernet.in

² Also affiliated to: Alcatel-Lucent India Limited, Bangalore, India
sourav10101976@gmail.com

Abstract. Non-linearity as well as randomness are essential for cryptographic applications. The Linear Cellular Automata (CA), particularly maximum length CA, are well known for generating excellent random sequences. However, till date, adequate research has not been done to generate maximal length Cellular Automata using non-linear rules; a fact that limits the application of CA in cryptography. This paper devises a method to generate non-linear Maximal Length Cellular Automata. It manipulates the number of clock cycles, based on inputs, in a maximum length additive CA and generates a series of non-linear boolean mappings. It shows that the bit streams generated in this manner are highly non-linear and pass all the statistical tests for randomness. These maximum length CA can be used as a non-linear primitive in cryptographic applications.

Keywords: Pseudo-Random Number Generator, Non-linearity, Maximum Length Cellular Automata, Cryptography.

1 Introduction

Cellular Automata (CA) are computational models that can perform complex computation with only local information. The simple structure of CA has attracted researchers and practitioners of different fields and has undergone rigorous theoretical and experimental analysis. The complex behavior of natural systems from diverse disciplines has been modeled in very simplistic manner by using CA. CA have also gained popularity due their simplicity in hardware implementations. CA based architectures are regular, cascadable and also with lesser interconnects which help VLSI design.

CA have been categorized into two broad categories: Additive/Linear CA where the local transformations of a CA have been driven by XOR/XNOR logic [1] and non-linear CA where there are no such restrictions. The additive CA have been analyzed using algebraic methods and have been used widely in the area of VLSI design and test. The applications of CA include pattern generation/recognition, biological computing, neural networks, cryptography to name a few.

Using Additive CA, it is possible to generate maximal length CA where all possible states of an n bit pattern lie in a single cycle. A lot of work has been done to generate VLSI test patterns using linear CA [3], [4], [5], [6] which give good randomness property. While the randomness property is necessary, the non-linearity is also a must for cryptography. Otherwise, the key in a cryptographic application can be recovered by solving a set of linear equations. Hence we need a non-linear CA to utilize it effectively in the cryptographic applications. Wolfram has introduced CA as random sequence generator, which is useful for cryptography, and used rule 30 for non-linearity [10], [11]. However, rule 30 does not give maximum length sequences. Using this fact, Meier and Staffelbach [12] have found that in some cases the change in seed does not change the output sequences. Hence, they could attack the CA based non-linear sequence generator easily. This provides the need for a non-linear CA which is of maximum length. However, it is not possible to generate maximum length CA using non-linear rules.

In this paper, we address the problem of non-existence of non-linear maximal length CA and vice versa, that is maximal length CA not having non-linearity. We use additive CA for maximal length CA and derive the non-linearity by varying the clock cycles based on input. We show that we can generate excellent non-linearity using such a scheme. Also, by selecting a maximal length additive CA, the proposed construction can have all the possible states in a single cycle.

This paper is organized as follows: Section 2 provides an introduction to Cellular Automata and two related concepts. The third section describes how to generate non-linear boolean mappings using CA. It gives the theoretical basis with some examples and the effect of different seeds on the construction. The section 4 provides the experimental results. The section 5 gives the statistical evaluation for randomness performed for these bits using NIST statistical test suit. Finally, the section 6 describes the hardware implementation and the software implementation for proposed construction of non-linear CA.

2 Preliminaries

In this section, we describe some basic theoretical foundations of the proposed work. We first describe some basics of Cellular Automata.

2.1 Basics of Cellular Automata

Wolfram [7] pioneered the study of CA as a mathematical model for self organizing statistical systems. The CA structure can be viewed as a lattice of cells where every cell can take values either 0 or 1. Each cell evolves in each time step depending on some combinational logic on itself and its neighbors as shown in figure [1] [1]. Such a CA is called *three-neighborhood CA*. The next state function for a three-neighborhood CA cell can be expressed as follows:

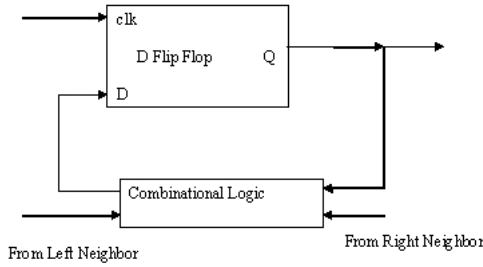


Fig. 1. A Cell of Cellular Automata

Say,

i : position of an individual cell in an one dimensional array.

t : time step.

$q_i(t)$: output state of the i -th cell at the t -th time step.

Then,

$$q_i(t + 1) = f(q_i(t), q_{i+1}(t), q_{i-1}(t)),$$

where f denotes the local transition function realized with a combination logic and is known as a rule of the CA.

A three neighborhood CA having two states (0 and 1) can have 2^3 distinct neighborhood configurations. For such a CA there can be a total of 2^{2^3} (256) distinct mappings from all those neighborhood configurations to the next state. Each mapping is called a "rule" of the CA. Among the rules, rule 90 and rule 150 are used to generate the maximum length CA. The combinational logic for the above rules can be given as:

$$\text{rule90} : q_i(t + 1) = q_{i+1}(t) \oplus q_{i-1}(t)$$

$$\text{rule150} : q_i(t + 1) = q_i(t) \oplus q_{i+1}(t) \oplus q_{i-1}(t)$$

If the rule of a CA involves XOR logic only it is called a linear rule and the corresponding CA is called a *linear CA* or an *additive CA*.

The rules with AND-OR combination logic are called *non-additive CA*. Non-additive Cellular Automata are non-linear in nature.

The state of an n-cell CA can be represented by its *characteristic polynomial*. The *characteristic matrix* of a CA operating over GF(2) is a matrix that describes the behavior of the CA. We can calculate the next state of the CA by multiplying the characteristic matrix by the present state of the CA. A characteristic matrix is constructed as:

$$T[i, j] = 1, \text{ if the next state of the } i\text{th cell depends on } j\text{th cell} \\ = 0, \text{ otherwise}$$

The associated characteristic polynomial can be obtained by constructing the matrix $[T] + x[I]$ where I is the identity matrix and then computing the

determinant of the resultant matrix. If $S(t)$ represents the state of the CA at the t -th time step then the state at the next time instant can be represented as:

$$S(t + 1) = [T]S(t) \text{ and } S(t + 2) = [T]^2S(t) \text{ and so on. So we can write:}$$

$$S(t + p) = [T]^pS(t)$$

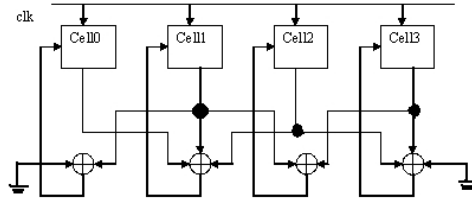


Fig. 2. Four Cell Maximum Length Cellular Automata Using 90,150,90,150 Rules

A *Group CA* is one in which each of the states has a single predecessor. A maximum length CA is a group CA with all non-zero states lying in a single cycle. It has been established that the maximum length cycle can be produced only if the characteristic polynomial is primitive as well as only if rule 90 and/or rule 150 is used to construct the CA (rule 90 = xor(left neighbor, right neighbor); rule 150 = xor(left neighbor, self, right neighbor)). A four cell maximum length CA has rules (90,150,90,150) and is shown in figure 2

2.2 Non-linearity

We call n tuple of elements from $GF(2)$ as a vector space, V_n . The n tuple of V_n can assume an integer value from $[0 \cdots 2^n - 1]$ and thus each element in V_n has an one to one correspondence with integers of this range. The Hamming weight of a vector x of n elements is the number of ones in x . If f and g are two functions in V_n then the Hamming distance of f and g are defined as $d(f, g) = \sum_{f(x) \neq g(x)} 1$, where the addition is over the integers. We call $h(x) = a_1x_1 \oplus a_2x_2 \oplus a_3x_3 \oplus \cdots \oplus a_nx_n \oplus c$ as the affine function where $a_i, c \in GF(2)$ and $x = (x_1, x_2, \cdots, x_n)$. If $c = 0$ the above function is called a linear function. Let $h_0, \cdots, h_{2^{n+1}-1}$ be the affine functions in V_n . The non-linearity of a function f is defined as $N_f = \min_{0, \dots, 2^{n+1}-1} d(f, h_i)$.

As studied in [13], the following Eq. 1 provides the non-linearity for the function f in V_n ,

$$N_f = 2^{n-1} - 0.5 \max \left| \sum_{u,x \in V_n} (-1)^{f(x)+u \cdot x} \right| \tag{1}$$

From the above equation, the maximum non-linearity that can be achieved is $2^{n-1} - 2^{0.5n-1}$.

However, since in this paper we have generated $n \times n$ bit non-linear boolean mappings, we should consider the linear combination of the coordinate functions [8]. In that case the Eq. 1 should be modified as in Eq. 2.

$$N_f = 2^{n-1} - 0.5max \left| \sum_{u,v,x \in V_n} (-1)^{v \cdot f(x) + u \cdot x} \right| \tag{2}$$

Here, the maximum non-linearity is upper bounded by $2^{n-1} - 2^{0.5(n-1)}$.

2.3 Algebraic Normal Form

The algebraic normal form is a representation of boolean functions that uses only AND and XOR functions. Let F_2^n be the vector space defined by n-vectors $\mathbf{x} = (x_1, x_2, \dots, x_n)$, where $x_i \in F_2$, i.e. each of the n elements has either value 0 or 1 and computations are defined modulo 2. A Boolean function f of n variables is simply a mapping $f: F_2^n \mapsto F_2$. There are exactly 2^{2^n} distinct Boolean functions of n variables, each uniquely defined by its truth table. The algebraic normal form, $\hat{f}: F_2^n \mapsto F_2$, is defined by the function:

$$\hat{f} = \sum_{a \in F_2^n} f(a) \prod_{i=1}^n x_i^{a_i}$$

There is a unique algebraic normal form for all boolean functions f . The *algebraic degree* $deg(f)$ is the maximum Hamming weight \mathbf{x} that satisfies $\hat{f}(\mathbf{x}) = 1$; this is equivalent to the length of the longest monomial (most variables) in the polynomial representation of f .

3 Generation of Non-linearity

Any CA transformation takes two input parameters. The first one is the seed of the CA and the second one is the number of clock cycles that needs to be run. The relationship between the input seed and the output is completely linear as it involves only XOR operations. However, a very little attention has been given for the relationship between the number of cycles and the output of the CA. We have found that the relationship between the number of cycles and the output is highly non-linear. Also it spreads across statistically well for different inputs.

3.1 CA Based Transformation Function

If n denotes the seed, m denotes the number of cycles and y denotes the output of a CA transformation, then, any CA transformation can be expressed as:

$$y = T^m * (n) \tag{3}$$

where T (constant) is the characteristic matrix of the CA.

If we keep the seed constant and vary the number of cycles based on input x , the CA transformation becomes:

$$y = T^{(x)} * n \text{ mod } p(x) \tag{4}$$

where, $p(x)$ denotes the generator polynomial that gives rise to the maximum length CA. Hence the output varies exponentially with input giving rise to non-linearity.

Example 1

The above theory can be clear if we take a simple four cell maximum length CA having rule vector $\langle 90, 150, 90, 150 \rangle$ (for a generator polynomial $X^4 + X + 1$) and calculate its algebraic normal form. Consider a seed $\langle 1, 0, 1, 0 \rangle$. Let us denote the input vector as $\mathbf{x} = (x_1, x_2, x_3, x_4)$. This input vector will determine the number of cycles to be run for the CA transformation. Let, the vector $\mathbf{y} = (y_1, y_2, y_3, y_4)$ denote the corresponding outputs of the CA after running \mathbf{x} number of cycles. Also for an all zero input the output is hard-coded to zero.

Table 1. Generated 4×4 Boolean Mapping with Initial Seed 1010

INPUT Clk Cycles	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OUTPUT (Decimal Form)	0	1	3	6	11	2	5	13	9	7	8	4	14	15	12	10

The table 1 shows the input cycles vs. output of the CA mapping. The corresponding algebraic normal form is given below; all operations are performed on $GF(2)$ which means the '+' operations are equivalent to XOR operations.

$$\begin{aligned}
 y_1 &= x_0 + x_1 + x_2 + x_1x_2 + x_0x_1x_2 + x_3 + x_0x_3 + x_0x_2x_3 \\
 y_2 &= x_1 + x_2 + x_0x_3 + x_1x_3 + x_0x_1x_3 + x_0x_2x_3 + x_1x_2x_3 \\
 y_3 &= x_0x_1 + x_1x_2 + x_0x_1x_2 + x_0x_3 + x_0x_1x_3 + x_2x_3 + x_0x_2x_3 + x_1x_2x_3 \\
 y_4 &= x_2 + x_0x_2 + x_1x_2 + x_3 + x_0x_3 + x_2x_3 + x_1x_2x_3
 \end{aligned}$$

The above representation of algebraic normal form shows that each output bit is dependent on all the input bits. It can also be observed that the algebraic degree for each output bit is three. So the relationship between the number of cycles and the output of the CA is highly non-linear and has got very good statistical properties.

Example 2

Next consider the seed $\langle 1, 1, 1, 1 \rangle$. Table 2 shows a mapping thus generated between the number of cycles and the output of the CA.

Table 2. Generated 4×4 Boolean Mapping with Initial Seed 1111

INPUT Clk Cycles	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OUTPUT (Decimal Form)	0	12	10	1	3	6	11	2	5	13	9	7	8	4	14	15

The corresponding algebraic normal form is shown below; all operations are performed on $GF(2)$ which means the '+' operations are equivalent to XOR operations.

$$\begin{aligned}
 y_1 &= x_0x_1 + x_2 + x_0x_2 + x_0x_1x_2 + x_3 + x_0x_1x_3 + x_0x_2x_3 \\
 y_2 &= x_1 + x_0x_1 + x_2 + x_1x_2 + x_0x_1x_2 + x_1x_3 + x_2x_3 \\
 y_3 &= x_0 + x_0x_1 + x_3 + x_0x_3 + x_1x_3 + x_2x_3 + x_0x_2x_3 \\
 y_4 &= x_0 + x_1 + x_0x_2 + x_0x_1x_2 + x_2x_3 + x_0x_2x_3 + x_1x_2x_3
 \end{aligned}$$

Here also, the above representation of algebraic normal form shows that each output bit is dependent on all the input bits. It can also be observed that the algebraic degree for each output bit is three.

3.2 The Effect of Seed

In the above construction, different seeds will lead to a different starting point of the same cycle, which is expected from a maximum length additive CA. However, it is possible to get a completely new Input/Output mapping, when the input vector is the number of clock cycles, with each different seed. This is obvious from the two examples given in the previous subsection. There, we can see that two different seeds 1010 and 1111 produce two different Input/Output mappings and hence produce a different algebraic normal form in the output expressions. However, if we check carefully, the outputs in both the cases, lie in the same state machine.

4 The Non-linearity of the Construction

In this section, we examine the non-linearity achieved through the non-linear maximum length Cellular Automata as described in the previous sections. If we put the output expression of the boolean mapping Eq. 4 in the non-linearity Eq. 2, the resulting expression is shown in Eq. 5.

$$N_f = 2^{n-1} - 0.5max \left| \sum_{u,v,x \in V_n} (-1)^{v \cdot (T^{(x)} * n \text{ mod } p(x)) + u \cdot x} \right| \tag{5}$$

The actual non-linearity would depend on the choice of T matrix which, in turn, depends on the Rule vector chosen for the construction. Hence, theoretically, it gives a very good indication that the proposed maximum length non-linear CA will provide excellent non-linearity.

To show that the proposed construction can provide non-linearity, we provide the experimental results of non-linearity in this section. Due to space and system limitation in calculating the non-linearity values, we give the non-linearity values for CAs of length 4 to 16. The table 3 shows the non-linearity of each output bits. In these calculations, the maximum length additive CAs are used with rule vectors as defined in 9. The seeds used for each of these CAs are alternating ones and zeros starting with one. The first column of the table gives the number of cells of the CA. The last column provides the maximum possible non-linearity

Table 3. Non-linearity of Each Output bit

No of Cells	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit 12	Bit 13	Bit 14	Bit 15	Bit 16	Theo. Max
4	4	4	4	4	-	-	-	-	-	-	-	-	-	-	-	-	5
5	10	12	10	10	10	-	-	-	-	-	-	-	-	-	-	-	12
6	24	24	22	22	20	20	-	-	-	-	-	-	-	-	-	-	26
7	52	48	52	50	52	52	50	-	-	-	-	-	-	-	-	-	56
8	108	106	108	110	108	106	108	108	-	-	-	-	-	-	-	-	116
9	216	226	224	224	218	220	216	218	220	-	-	-	-	-	-	-	240
10	470	460	466	470	464	470	468	466	470	462	-	-	-	-	-	-	489
11	946	958	948	940	930	932	938	952	942	936	950	-	-	-	-	-	992
12	1944	1934	1938	1938	1922	1948	1954	1940	1922	1944	1934	1936	-	-	-	-	2002
13	3938	3916	3940	3914	3936	3910	3928	3942	3914	3954	3944	3938	3914	-	-	-	4032
14	7998	7902	7888	7828	7872	7902	7898	7808	7902	7916	7922	7892	7872	7878	-	-	8101
15	15998	16012	15896	16000	15998	16026	16026	15984	15990	15998	16040	16038	16016	16036	16012	-	16256
16	32218	32240	32144	32240	32188	32240	32278	32220	32224	32126	32224	32248	32184	32274	32252	32216	32586

according to Chabaud-Vaudenay bound [8]. We can see that the non-linearity value in all the cases are quite close the maximum. In the table, the theoretical maximum was taken as the nearest integer less than the value obtained by the formula.

5 Statistical Tests

To evaluate the mappings for statistical randomness, we have used NIST [2] statistical test tool. The NIST Statistical Test Suit was used to evaluate AES candidates for randomness. To test the outputs against randomness, we have used a CA of length 16 and the output for all the 2^{16} inputs were fed to the tool. This test tool contain sixteen statistical tests to evaluate a random number generator. Table 4 shows the result of the statistical tests. The results are shown for each of the output bits and all the output bits taken together. Three tests were not run for individual output bits due to insufficient data size in a sixteen bits CA. However, just to run these three tests, we used a 32 bit CA where these three tests also passed.

6 Implementation

In this section, we provide both the hardware implementation and the software implementation details of the non-linear CA.

6.1 Hardware Implementation

The hardware implementation of the maximum length non-linear cellular automata is given in figure 3. An n cell maximum length CA is loaded with a constant initial seed. The actual input is pre-processed to control the clock signal. The input is compared with the output of an n bit counter by a comparator circuit. When the input is equal to the counter value, comparator output

Table 4. Statistical Tests Summary

Test	Single Bit	All Bits
Frequency MonoBit	Pass	Pass
Block Frequency	Pass	Pass
Runs	Pass	Pass
Longest Run of ones in a block	Pass	Pass
Rank	Pass	Pass
DFT	Pass	Pass
Non-overlapping template matching	Pass	Pass
overlapping template matching	Pass	Pass
Universal Statistical Test	Not Run	Pass
Lempel Ziv	Pass	Pass
Linear Complexity	Pass	Pass
Serial Test	Pass	Pass
Approximate Entropy	Pass	Pass
CUSUM	Pass	Pass
Random Excursion	Not Run	Pass
Random Excursion Variant	Not Run	Pass

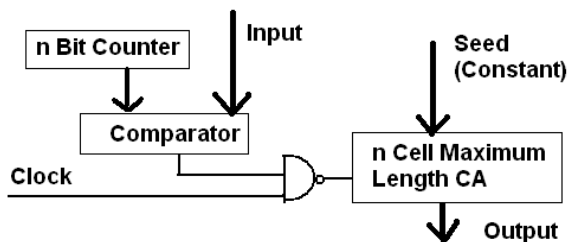


Fig. 3. Hardware Implementation

becomes 0 which stops the clock signal by the use of a NAND gate. When the counter value is less than the input, the comparator output remains at logic 1 and thus the clock signal is fed to the CA.

6.2 Software Implementation

The software implementation of the non-linear construction is to run the CA in a loop till the input value. A simple C function for the software implementation is given below. Note that, the below implementation will work only for CA of length till 31. However, this code can easily be extrapolated to handle more number of bits. *input* is the input to the function that will be transformed non-linearly to give the output.

1. `#define RULE /* define the rule (in decimal form)*/`
2. `#define INITIAL_SEED /* define the initial seed in decimal form*/`

```

3. #define MAX_INPUT /* (2n - 1) where n is the length of the CA*/
4. int i;
5. int seed=INITIAL_SEED;
6. if (input==0) return 0;
7. for (i=0;i<input;i++)
8. seed=((seed<=1) & MAX_INPUT) ^ (seed & RULE) ^ ((seed>=1) &
MAX_INPUT);
9. return seed;

```

7 Conclusion

In this paper, we have shown a way to generate non-linearity using CA. This way, we can generate maximal length CAs with excellent non-linearity. The highly non-linear maximum length CA can be used in cryptography.

References

1. Chaudhury, P.P., Chowdhury, D.R., Nandi, S., Chattopadhyay, S.: Additive Cellular Automata Theory and Application, vol. 1. IEEE Computer Society Press, Los Alamitos
2. NIST Statistical Test Suit, <http://csrc.nist.gov/rng/>
3. Serra, M., Slater, T., Muzio, J.C., Miller, D.M.: Analysis of One Dimensional Cellular Automata and their Aliasing Probabilities. IEEE Trans. on CAD 9(7), 767–778 (1990)
4. Tomassini, M., Sipper, M., Perrenoud, M.: On the Generation of High-Quality Random Numbers by Two-dimensional Cellular Automata. IEEE Trans. on Computers 49(10), 1146–1151 (2000)
5. Nandi, S., et al.: Analysis of Periodic and Intermediate Boundary 90/150 Cellular Automata. IEEE Trans. on Computers 45(1), 1–12 (1996)
6. Hortencius, P.D., McLeod, R.D., Pries, W., Miller, D.M., Card, H.C.: Cellular Automata based Pseudo-random Number Generators for Built-in Self-Test. IEEE Trans. on CAD 8(8), 842–859 (1989)
7. Wolfram, S.: A New Kind of Science. Wolfram Media, Champaign (2002)
8. Chabaud, F., Vaudenay, S.: Links between differential and linear cryptanalysis. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 356–365. Springer, Heidelberg (1995)
9. Cattell, K., Zhang, S.: Minimal Cost One-Dimensional Linear Hybrid Cellular Automata of Degree Through 500. Journal Of Electronic Testing: Theory and Applications 6, 255–258 (1995)
10. Wolfram, S.: Random Sequence Generation by Cellular Automata. Advances in Applied Mathematics 7, 123 (1986)
11. Wolfram, S.: Cryptography with Cellular Automata. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 429–432. Springer, Heidelberg (1986)
12. Meier, W., Staffelbach, O.: Anaysis of Pseudo Random Sequences Generated by Cellular Automata. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 186–199. Springer, Heidelberg (1991)
13. Nyberg, K.: On the construction of highly nonlinear permutations. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 92–98. Springer, Heidelberg (1993)

Chaotic Cellular Automata with Cryptographic Application^{*}

Amparo Fúster-Sabater¹ and Pino Caballero-Gil²

¹ Institute of Applied Physics, C.S.I.C.
Serrano 144, 28006 Madrid, Spain
`amparo@iec.csic.es`

² Faculty of Maths, D.E.I.O.C.
University of La Laguna, 38271 Tenerife, Spain
`pcaballe@ull.es`

Abstract. In this paper, a method of generating cryptographic sequences based on discrete linear chaotic cellular automata is presented. The importance of the proposal is due to the fact that such cryptographic sequences are also output sequences of a nonlinear keystream generator known as Generalized Self-Shrinking Generator. Moreover, such a keystream generator is still considered secure in symmetric cryptography. Thus, it must be noticed that the linearity of the proposed chaotic model based on additive one-dimensional cellular automata might be used to mount a cryptanalytic attack against such a nonlinear generator.

Keywords: Cellular automata, cryptography, chaotic system, sequence generator.

1 Introduction

Confidentiality in sensitive information is a crucial feature. Such a quality makes use of an encryption function currently called *cipher* that converts the original message (*plaintext*) into the ciphered message (*ciphertext*). Symmetric cryptography or secret-key cryptography is usually divided into two large classes [17]: block ciphers and stream ciphers depending on whether the encryption function is applied to a bit block or to each individual bit, respectively.

At present, stream ciphers are the fastest among the encryption procedures so they are implemented in many technological applications e.g. mobile phones (GSM communications) or encryption procedures in Bluetooth specifications. Stream ciphers are designed to generate from a short seed, the *key*, a long sequence of pseudorandom bits, the *keystream sequence*. Such a sequence is XORed with the plaintext (in emission) in order to obtain the ciphertext or with the ciphertext (in reception) in order to recover the plaintext. Security of a stream

* This work was supported in part by CDTI (Spain) and the companies INDRA, Unión Fenosa, Tecnobit, Visual Tools, Brainstorm, SAC and Technosafe under Project Cenit-HESPERIA; by Ministry of Science and Innovation and European FEDER Fund under Project TIN2008-02236/TSI.

cipher resides in the characteristics of the keystream sequence: long period, good statistical properties and high linear complexity [8].

In order to design new keystream generators, in the last decades many researchers have exploited the close relationship between chaos and cryptography. See, for instances, [1], [4], [6], [12] and [16]. In fact, the reason of such a relationship is that many properties of chaotic systems have their corresponding counterparts in traditional cryptographic systems. Among the first applications of chaotic structures or, more precisely, chaotic Cellular Automata (CA) to symmetric cryptography we can enumerate the references [13], [20] and [22]. However, other chaotic cryptographic systems have been rather disappointing since various cryptanalysis have revealed different drawbacks inherent to such schemes [7], [15], [18] and [23]. Among the most recent contributions to the relationship chaos-cryptography, we find the reference [19] that analyzes the evolution of nonlinear CA generating desirable pseudorandom number sequences.

Traditionally, keystream generators make use of maximal-length Linear Feedback Shift Registers (LFSRs) [10] whose output sequences (the PN-sequences) are combined in a nonlinear way in order to produce the desired keystream sequences. Nevertheless, it is a well known fact that certain linear chaotic CA [3] generate exactly the same PN-sequences obtained from maximal-length LFSRs. In this way, multiple cryptographic keystream generators designed in terms of LFSRs can be also expressed in terms of chaotic CA. Indeed, in this work it is shown that a wide class of LFSR-based nonlinear generators known as Generalized Self-Shrinking Generators (GSSGs) [11] can be modelled in terms of linear CA. Since linearity in the behaviour of a cipher may be considered as the end of its security, the result here shown may be interpreted as a proof of cryptographic weakness for the GSSG family.

Analyzing chaotic systems as generators of cryptographic sequences allowed us to find an alternative way to produce the same sequences as those of the GSSG family. Thus, the use of CA as linear chaotic models of the nonlinear GSSG is believed to be an approach to its cryptanalysis. In addition, the building procedure of linear CA-based models for GSSGs can be implemented by means of simple FPGA logic.

The paper is organized as follows. In Section 2, the specific type of additive one-dimensional linear chaotic CA used in this work has been introduced. Description and characteristics of the GSSG are given in section 3. Chaotic modelling of GSSG exploiting its relationship with previous CA is carried out in Section 4. Finally, conclusions in Section 5 end the paper.

2 Cellular Automata

One-dimensional cellular automata can be described as N -cell registers [21], whose cell contents are updated at the same time according to a particular rule. That is a k -variable function denoted by Φ . If the function Φ is a linear function so is the cellular automaton. When k input variables are considered, then there is a total of 2^k different binary neighbor configurations.

Therefore, for cellular automata with binary content cells there can be up to 2^{2^k} different mappings to the next state. Moreover, if $k = 2r + 1$, then the next state x_i^{t+1} of the cell x_i^t depends on the current state of k neighbor cells $x_i^{t+1} = \Phi(x_{i-r}^t, \dots, x_i^t, \dots, x_{i+r}^t)$ ($i = 1, \dots, n$).

CA are called *uniform* when all cells evolve under the same rule while CA are called *hybrid* when different cells evolve under different rules. At the ends of the array, two different boundary conditions are possible: *null automata* when cells with permanent null content are supposed adjacent to the extreme cells or *periodic automata* when extreme cells are supposed adjacent. In this paper, all the considered automata will be one-dimensional null hybrid CA with $k = 3$ and linear rules 90 and 150. These rules are described as follows:

<p style="text-align: center;">Rule 90</p> $x_i^{t+1} = x_{i-1}^t + x_{i+1}^t$ <p>111 110 101 100 011 010 001 000</p> <p>0 1 0 1 1 0 1 0</p> <p>01011010 (binary) = 90 (decimal)</p>	<p style="text-align: center;">Rule 150</p> $x_i^{t+1} = x_{i-1}^t + x_i^t + x_{i+1}^t$ <p>111 110 101 100 011 010 001 000</p> <p>1 0 0 1 0 1 1 0</p> <p>10010110 (binary) = 150 (decimal)</p>
--	--

Remark that the names rule 90 and rule 150 derive from the decimal values of their next-state functions.

Table 1. An one-dimensional null hybrid linear cellular automaton of $N = 10$ cells with rule 90 and rule 150 starting at a given initial state

90	150	150	150	90	90	150	150	150	90
0	0	0	1	1	1	0	1	1	0
0	0	1	0	0	1	0	0	0	1
0	1	1	1	1	0	1	0	1	0
1	0	1	1	1	0	1	0	1	1
0	0	0	1	1	0	1	0	0	1
0	0	1	0	1	0	1	1	1	0
0	1	1	0	0	0	0	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Both rules belong to Class III (chaotic behaviour) in Wolfram’s terminology [21]. For an one-dimensional null hybrid cellular automaton of $N = 10$ cells, configuration rules (90, 150, 150, 150, 90, 90, 150, 150, 150, 90) and initial state (0, 0, 0, 1, 1, 1, 0, 1, 1, 0), Table 1 illustrates the formation of its output sequences, sequences read vertically. Within this CA class, it is easy to determine the number of different sequences generated by a particular automaton, the distinct periods and linear complexities [17] of such sequences as well as the number of

different sequences associated with each period and linear complexity. See [9] and [2] for more details.

A natural form of representation for a 90/150 automaton is by means of a binary N -tuple (*rule vector*), notated $\Delta_N = (d_1, d_2, \dots, d_N)$, where $d_i = 0$ if the i -th cell satisfies the rule 90 while $d_i = 1$ if the i -th cell satisfies the rule 150. This rule vector allows one to determine the N -degree characteristic polynomial $P_N(x)$ of such an automaton, that is the polynomial that defines the recurrence relationship of the generated sequences, see [3]. In fact, $P_N(x)$ can be easily obtained from its rule vector as $P_N(x) = (x + d_1)(x + d_2) \cdots (x + d_N)$.

In addition, $P_N(x)$ is also the characteristic polynomial of the output sequences [17], that is to say it is the polynomial that express a term of any output sequence as a linear combination of the previous terms of such a sequence. Note that all output binary sequences produced with any rule of a 90/150 automaton in the same state cycle have the same period and linear complexity [9].

3 Generalized Self-shrinking Generator

The generalized self-shrinking generator is a generator of pseudorandom sequences recently introduced by Hu and Xiao in [11]. Such a generator may be seen as the generalization of both the shrinking generator introduced by Copper-smith, Krawczyk and Mansour in [5], and the self-shrinking generator defined by Meier and Staffelbach in [14]. The GSSG produces cryptographic sequences with long periods, good correlation features, excellent run distribution, balancedness, simplicity of implementation, etc. Furthermore, no practical attack against it is known till now. Consequently, it is still considered a secure cryptographic generator.

The GSSG can be formally defined as follows [11].

Definition 1. *Let $\{a_n\}$ be a PN-sequence over $GF(2)$ with period $2^L - 1$ generated with a LFSR of primitive characteristic polynomial of degree L . Let G be an L -dimensional binary vector:*

$$G = (g_0, g_1, g_2, \dots, g_{L-1}) \in GF(2)^L.$$

Let $\{v_n\}$ be a sequence defined as:

$$v_n = g_0 a_n \oplus g_1 a_{n-1} \oplus g_2 a_{n-2} \oplus \cdots \oplus g_{L-1} a_{n-L+1},$$

where the sub-indexes of the sequence $\{a_n\}$ are reduced mod $2^L - 1$ and the symbol \oplus represents the XOR logic operation.

For $n \geq 0$, let the decimation rule be:

- *If $a_n = 1$, then v_n is an output bit.*
- *If $a_n = 0$, then v_n is discarded, and no output bit is produced.*

In this way, an output sequence $b_0 b_1 b_2 \dots$ is generated. Such a sequence, denoted by $\{b_n\}$ or $\{b(G)\}$, is called generalized self-shrinking sequence associated with G .

The previously defined family of generalized self-shrinking sequences can be easily generated with a LFSR, as a result of the following two simple steps:

1. Two sequences are synchronously generated: the PN-sequence $\{a_n\}$, and the shifted version of such a sequence denoted by $\{v_n\}$.
2. The output generalized self-shrinking sequence is produced by applying the described decimation rule on both sequences $\{a_n\}$ and $\{v_n\}$.

It is important to remark that the sequence $\{v_n\}$ is simply a shifted version of the sequence $\{a_n\}$, and consequently the resulting family of generalized self-shrinking sequences includes the self-shrinking sequence as a special case.

When G ranges over $GF(2)^L$, then $\{v_n\}$ corresponds to the $2^L - 1$ possible shifts of $\{a_n\}$. In addition, the set of sequences denoted by $B(a) = \{\{b(G)\}, G \in GF(2)^L\}$ is the family of generalized self-shrinking sequences based on the PN-sequence $\{a_n\}$.

Let us see a simple example.

Example 1. For the 4-degree m -sequence $\{a_n\} = \{111101011\ 001000\}$ whose characteristic polynomial is $x^4 + x^3 + 1$, we get 16 generalized self-shrinking sequences based on the sequence $\{a_n\}$ (see [11]):

1. $G = (0000), \{b(G)\} = 00000000 \sim$
2. $G = (1000), \{b(G)\} = 11111111 \sim$
3. $G = (0100), \{b(G)\} = 01110010 \sim$
4. $G = (1100), \{b(G)\} = 10001101 \sim$
5. $G = (0010), \{b(G)\} = 00111100 \sim$
6. $G = (1010), \{b(G)\} = 11000011 \sim$
7. $G = (0110), \{b(G)\} = 01001110 \sim$
8. $G = (1110), \{b(G)\} = 10110001 \sim$
9. $G = (0001), \{b(G)\} = 00011011 \sim$
10. $G = (1001), \{b(G)\} = 11100100 \sim$
11. $G = (0101), \{b(G)\} = 01101001 \sim$
12. $G = (1101), \{b(G)\} = 10010110 \sim$
13. $G = (0011), \{b(G)\} = 00100111 \sim$
14. $G = (1011), \{b(G)\} = 11011000 \sim$
15. $G = (0111), \{b(G)\} = 01010101 \sim$
16. $G = (1111), \{b(G)\} = 10101010 \sim$

Note that the above generated sequences are 16 sequences, but not all are different. In fact, there are exactly 7 different sequences. In particular, if we refer to them by the decimal value of the binary representation G read from left to right, we have that sequences 4 and 5 are shifted versions of the same sequence, and the same applies for sequences 10 and 11 and to sequences 14 and 15. At the same time, sequences 2, 6, 9 and 12 correspond to a unique sequence and sequences 3, 7, 8 and 13 also correspond to another unique sequence.

4 CA-Based Model of the GSSG

Table 2 contains some experimental results of an implementation of the GSSG, which show a remarkable property. For every LFSR of length L , the corresponding GSSG produces always one all-zero sequence with characteristic polynomial 1, one all-one sequence with characteristic polynomial $(x + 1)$, two sequences of period 2 and characteristic polynomial $(x + 1)^2$, and 2^j sequences with characteristic polynomials $(x + 1)^{2^{L-1}-2(L-1)+j+1}$ with $j = 2, 3, \dots, L - 1$, respectively.

Table 2. Number of GSS sequences with characteristic polynomial of the form $(x + 1)^p$

L	3	3	3	3	4	4	4	4	4	5	5	5	5	5	5	6	6	6	6	6	6	6
No.	1	1	2	4	1	1	2	4	8	1	1	2	4	8	16	1	1	2	4	8	16	32
p	0	1	2	3	0	1	2	5	6	0	1	2	11	12	13	0	1	2	25	26	27	28

Hence, the characteristic polynomial of every GSS sequence produced with a LFSR of length L is always of the form $(x + 1)^p$, $0 \leq p < 2^{L-1}$. In this way, since such a polynomial is a unique factor multiplied by itself p times, it seems quite natural to construct the corresponding automaton that generates the same output sequences by concatenating p times a basic 90/150 automaton associated to that of characteristic polynomial $(x + 1)$. The procedure of concatenation is based on the following general result.

Theorem 1. *Let B be a 90/150 cellular automaton of length L , characteristic polynomial $P(x)$ and rule vector $\Delta_L = (d_1, d_2, \dots, d_{L-1}, d_L)$. Let B^* be the reversal version of B , with rule vector $\Delta_L^* = (d_L, d_{L-1}, \dots, d_2, d_1)$ and the same length and polynomial as B . Then, the $2L$ -tuple $(d_1, d_2, \dots, \overline{d_L}, \overline{d_L}, \dots, d_2, d_1)$ (being $\overline{d_L}$ the complementation of d_L) is the rule vector of a 90/150 cellular automaton of length $2L$ and characteristic polynomial $P(x)^2$.*

A proof of this theorem by the same authors can be found in [9].

Such a result can be iterated a number of times for successive characteristic polynomials and rule vectors:

$$\begin{aligned}
 P_L(x) = P(x) & \longleftrightarrow \Delta_L = (d_1, d_2, \dots, \overline{d_L}) \\
 P_{2L}(x) = P(x)^2 & \longleftrightarrow \Delta_{2L} = (d_1, d_2, \dots, \overline{d_L}, \overline{d_L}, \dots, d_2, d_1) \\
 P_{2^2L}(x) = P(x)^{2^2} & \longleftrightarrow \Delta_{2^2L} = (d_1, \dots, \overline{d_L}, \overline{d_L}, \dots, \overline{d_1}, \overline{d_1}, \dots, \overline{d_L}, \overline{d_L}, \dots, d_1) \\
 \vdots & \longleftrightarrow \vdots
 \end{aligned}$$

In this way, the concatenation of an automaton (with the least significant bit complemented) and its mirror image allows us to synthesize CA with known characteristic polynomials. Note that the basic automaton is concatenated with its reversal version after the complementation of the least significant rule. The successive applications of this result provide us with CA whose lengths are $L, 2L, 2^2L, 2^3L, \dots, 2^qL$, respectively. Also note that for every $P(x)$ there are two

basic CA $\Delta_L = (d_1, d_2, \dots, d_{L-1}, d_L)$ and $\Delta_L^* = (d_L, d_{L-1}, \dots, d_2, d_1)$ that may be used in the concatenation process.

It is remarkable the fact that the automaton $\Delta_{2^q L}$ includes all the previous sub-automata $\Delta_{2^s L}$ with $0 \leq s < q$, and consequently the automaton $\Delta_{2^q L}$ generates all the sequences whose characteristic polynomials are $P(x)^p$ with $1 \leq p \leq 2^q$. The choice of a particular state cycle determines the corresponding characteristic polynomial of its sequences.

In the specific case of the GSSG, as the automaton corresponding to $P(x) = (x + 1)$ is a simple rule 150, that is to say, $\Delta_1 = (1)$, the application of the previous result allows us to derive the following relationships between characteristic polynomials and rule vectors:

$$\begin{aligned}
 (x + 1) &\longleftrightarrow \Delta_1 = (1) \\
 (x + 1)^2 &\longleftrightarrow \Delta_2 = (0, 0) \\
 (x + 1)^4 &\longleftrightarrow \Delta_2 = (0, 1, 1, 0) \\
 (x + 1)^8 &\longleftrightarrow \Delta_8 = (0, 1, 1, 1, 1, 1, 1, 0) \\
 \vdots &\longleftrightarrow \vdots \\
 (x + 1)^{2^{L-1}} &\longleftrightarrow \Delta_{2^{L-1}} = (0, 1, 1, \dots, 1, 1, 0)
 \end{aligned}$$

In this way, rule vectors corresponding to 90/150 CA whose characteristic polynomials are squared powers of $(x + 1)$ may be easily obtained, so that the last rule vector corresponds to the required automaton for the GSSG based on a LFSR of length L .

Let us consider the GSSG based on the LFSR of length $L = 4$ and characteristic polynomial $x^4 + x + 1$ of the previous example. This GSSG produces sequences with the following periods, linear complexities and characteristic polynomials:

- 1 sequence with period 1, complexity 0 and polynomial 1,
- 1 sequence with period 1, complexity 1 and polynomial $(x + 1)$,
- 2 sequences with period 2, complexity 2 and polynomial $(x + 1)^2$,
- 4 sequences with period 8, complexity 5 and polynomial $(x + 1)^5$,
- 8 sequences with period 8, complexity 6 and polynomial $(x + 1)^6$.

According to the previous results, the 90/150 linear cellular automaton that generates such sequences has as rule vector $\Delta_8 = (0, 1, 1, 1, 1, 1, 1, 0)$.

Tables 3 and 4 show Δ_8 state transition, starting at 4 initial states $(0, 0, 0, 0, 1, 1, 1, 1)$, $(0, 0, 1, 0, 0, 0, 1, 0)$, $(1, 0, 0, 0, 1, 0, 0, 0)$ and $(0, 1, 1, 0, 0, 1, 1, 0)$. For the first initial state (Table 3), the four extreme rules produce the GSS sequences 4 and 5 whilst the four central rules produce the GSS sequences 10 and 11. For the second initial state (Table 3), we get that the four extreme rules lead to the GSS sequences 2, 6, 9 and 12 whilst the four central rules do not produce GSS sequences. The third initial state (Table 4) generates in the four extreme rules the GSS sequences 3, 7, 8 and 13 and again the four central rules do not produce any GSS sequence. Finally, the fourth initial state (Table 4) produces in all its rules the GSS sequences 14 and 15.

Note that the proposed automaton generates all the GSS sequences produced by all maximal-length LFSRs of length L . Therefore, the specific LFSR feedback

Table 3. 90/150 CA generating particular GSS Sequences

0 1 1 1 1 1 1 0	0 1 1 1 1 1 1 0
0 0 0 0 1 1 1 1	0 0 1 0 0 0 1 0
0 0 0 1 0 1 1 1	0 1 1 1 0 1 1 1
0 0 1 1 0 0 1 1	1 0 1 0 0 0 1 1
0 1 0 0 1 1 0 1	0 0 1 1 0 1 0 1
1 1 1 1 0 0 0 0	0 1 0 0 0 1 0 0
1 1 1 0 1 0 0 0	1 1 1 0 1 1 1 0
1 1 0 0 1 1 0 0	1 1 0 0 0 1 0 1
1 0 1 1 0 0 1 0	1 0 1 0 1 1 0 0

Table 4. 90/150 CA generating particular GSS Sequences

0 1 1 1 1 1 1 0	0 1 1 1 1 1 1 0
1 0 0 0 1 0 0 0	0 1 1 0 0 1 1 0
0 1 0 1 1 1 0 0	1 0 0 1 1 0 0 1
1 1 0 0 1 0 1 0	0 1 1 0 0 1 1 0
1 0 1 1 1 0 1 1	1 0 0 1 1 0 0 1
0 0 0 1 0 0 0 1	0 1 1 0 0 1 1 0
0 0 1 1 1 0 1 0	1 0 0 1 1 0 0 1
0 1 0 1 0 0 1 1	0 1 1 0 0 1 1 0
1 1 0 1 1 1 0 1	1 0 0 1 1 0 0 1

polynomial is not necessary for the generation as the automaton is exactly the same for any GSSG based in any maximal length LFSR of length L . Thus, the knowledge of such a polynomial, which is usually a part of the key, is not necessary for launching an attack based on the proposed CA-based model of the GSSG.

It is also remarkable that the proposed chaotic cellular automaton generates all the GSS sequences corresponding to LFSRs of lengths $< L$. That is to say, the longest automaton always includes all the GSS sequences corresponding to shorter automata by starting at symmetric initial states.

In this paper cryptographic generators conceived and designed as nonlinear generators have been linearized in terms of chaotic cellular automata. Consequently, the linearity of the proposed CA-based model of the GSSG might be exploited to mount a cryptanalytic attack based on the (partial) reconstruction of the keystream sequence from portions of intercepted sequence.

5 Conclusions

The Generalized Self-Shrinking Generator, which is a cryptographic generator conceived and designed as a nonlinear LFSR-based generator, has been here linearized thanks to the proposal of a simple and discrete time chaotic dynamical

system producing the same output sequences. The implementation of the proposed CA-based model is easy and very adequate for FPGA logic, what is useful for developments where time execution is relevant as in stream ciphers and in communication systems with high transmission rates.

Although generalized self-shrinking sequences are generated through irregular decimation on a LFSR, we have here shown that in practice they may be easily generated with a chaotic model based on additive one-dimensional cellular automata. This result establishes a link between nonlinear irregular decimation and linearity, which might be conveniently exploited in the cryptanalysis of such keystream generators.

References

1. Brown, R., Chua, L.: Clarifying chaos: examples and counterexamples. *Int. J. Bifurcat Chaos* 6(2), 219–242 (1996)
2. Caballero-Gil, P., Fúster-Sabater, A.: Using Linear Hybrid Cellular Automata to Attack the Shrinking Generator. *IEICE Transactions on Fundamentals of Electronics Communications and Computer E89-A*, 1166–1172 (2006)
3. Cattell, K., Muzio, J.C.: Synthesis of One-Dimensional Linear Hybrid Cellular Automata. *IEEE Trans. Computers-Aided Design* 15(3), 325–335 (1996)
4. Chaudhuri, P., Chowdhury, D., Nandi, S., Chatterjee, S.: Additive Cellular Automata. In: *Theory and Applications*, p. 1. IEEE Computer Society Press, Los Alamitos (1997)
5. Coppersmith, D., Krawczyk, H., Mansour, Y.: The Shrinking Generator. In: Stinson, D.R. (ed.) *CRYPTO 1993*. LNCS, vol. 773, pp. 22–39. Springer, Heidelberg (1993)
6. Dachselt, F., Schwarz, W.: Chaos and cryptography. *IEEE Trans. Circuits Syst.* 48(12), 1498–1509 (2001)
7. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. *Int. J. Bifurcat Chaos* 8(6), 1259–1284 (1998)
8. Fúster-Sabater, A., Caballero-Gil, P.: Cellular Automata in Cryptanalysis of Stream Ciphers. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) *ACRI 2006*. LNCS, vol. 4173, pp. 611–616. Springer, Heidelberg (2006)
9. Fúster-Sabater, A., Caballero-Gil, P.: Synthesis of Cryptographic Interleaved Sequences by Means of Linear Cellular Automata. *Applied Mathematics Letters* 22(10), 1518–1524 (2009)
10. Golomb, S.: *Shift-Register Sequences*. Aegean Park Press, Laguna Hill California (1982)
11. Hu, Y., Xiao, G.: Generalized Self-Shrinking Generator. *IEEE Trans. Inform. Theory* 50, 714–719 (2004)
12. Kocarev, L.: Chaos-based cryptography: a brief overview. *IEEE Circ. Syst.* 1, 6–21 (2001)
13. Meier, W., Staffelbach, O.: Analysis of Pseudo Random Sequences Generated by Cellular Automata. In: Davies, D.W. (ed.) *EUROCRYPT 1991*. LNCS, vol. 547, pp. 186–199. Springer, Heidelberg (1991)
14. Meier, W., Staffelbach, O.: The Self-Shrinking Generator. In: De Santis, A. (ed.) *EUROCRYPT 1994*. LNCS, vol. 950, pp. 205–214. Springer, Heidelberg (1995)
15. Parker, A., Short, K.: Reconstructing the keystream from a chaotic encryption scheme. *IEEE Trans. Circuits Syst. I* 48(5), 104–112 (2001)

16. Pradipta, M., Chandrama, S., Niloy, G., Biplab, S., Chaudhuri, P.: Theory and Application of Cellular Automata for Pattern Classification. *Fundamenta Informaticae* 58(3-4), 321–354 (2003)
17. Rueppel, R.: Stream Ciphers. In: Simmons, G.J. (ed.) *Contemporary Cryptology, The Science of Information*, pp. 65–134. IEEE Press, Los Alamitos (1992)
18. Tomassini, M., Perrenoud, M.: Cryptography with cellular automata. *Applied Soft Computing* 1(2), 151–160 (2001)
19. Tan, S.K., Guan, S.U.: Evolving cellular automata to generate nonlinear sequences with desirable properties. *Applied Soft Computing* 7(3), 1131–1134 (2007)
20. Urías, J., Ugalde, E., Salazar, G.: A cryptosystem based on cellular automata. *Chaos* 8(4), 819–822 (1998)
21. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)
22. Wolfram, S.: Cryptography with Cellular Automata. In: Williams, H.C. (ed.) *CRYPTO 1985. LNCS*, vol. 218, pp. 429–432. Springer, Heidelberg (1986)
23. Zhou, C., Lai, C.H.: Extracting messages masked by chaotic signals of time-delay systems. *Phys. Rev. E* 60, 320–323 (1999)

d-Monomial Tests of Nonlinear Cellular Automata for Cryptographic Design

Sandip Karmakar, Debdeep Mukhopadhyay, and Dipanwita Roy Chowdhury

Department of Computer Science and Engineering,
Indian Institute of Technology, Kharagpur,
India

Abstract. Pseudorandom generation is a key to any cryptographic application. Linear Cellular Automata are known as good pseudorandom generators. However, for cryptographic applications nonlinearity is essential for its security. But, nonlinear Cellular Automaton shows high correlation between the input to the automaton and its generated sequence. Hence, for cryptography Cellular Automata rules need to be nonlinear as well as satisfy additional properties. With this motivation, in this paper, we analyze nonlinear Cellular Automata with a newly developed statistical measure called *d*-monomial test. Finally, we propose a process of *d*-monomial characteristics addition to get cryptographically suitable Cellular Automata.

1 Introduction

Cellular Automata are a self-evolving system of cells which updates itself per cycle following a rule embedded into it. Linear Cellular Automaton (CA) is known for its ability to generate pseudorandom sequences needed for various applications like VLSI testing and coding theory [11]. Several researchers have attempted to apply the pseudorandomness of CA to cryptography. The cryptanalysis of linear CA based cryptographic techniques [4] show that nonlinearity is needed for cryptographic applications. However, nonlinear CA shows high correlation. The 3-neighbourhood nonlinear rule 30 CA has long been considered a good pseudo-random generator and studied for cryptography [10]. It passed various statistical tests for pseudorandomness with good results, until Willi Meier and Othmar Staffelbach proposed an attack, exploiting its high correlation, on pseudorandom sequences generated by rule 30 CA [6], which would break any such system of 300 cells with a complexity of about 2^{19} . Another attack on rule 30 CA is also reported in [5]. These findings show that for cryptography, the data stream generated by CA needs to satisfy additional properties.

In this paper, we analyze the CA by modeling its rule as a Boolean function relating output bits with input bits. Parameters like nonlinearity, balancedness, resiliency and algebraic degree are known to be important for the cryptographic analysis of Boolean functions [7, 8]. Recently, *d*-monomial tests [3] on cryptographic Boolean functions have gained attention. An extension of *d*-monomial test proposed in [2] serves as another important tool in analyzing cryptographic

Boolean functions. In [7] and [8] some of the stated cryptographic properties of 3 and 4 neighbourhood CA rules are analyzed for a single iteration of the CA. However, multiple iterations of the CA are not considered.

In this work, we explore CA over multiple iterations. At each iteration of the CA, the relationship between the input and output of the CA is represented by a Boolean function. Subsequently, we perform d -monomial tests on such Boolean functions and investigate how the test performs with iterations. It can be mentioned that, uniform and hybrid CA have not been investigated in perspective of cryptographic suitability in this direction before. Following the experimental results, we derive few general conclusions about choice of rules in uniform or hybrid CA to expect certain cryptographic advantages. We expect the findings will also help in analysis of non-linear CA, in general.

This paper is organized as follows. Following the introduction, section 2 presents basic definitions and notations regarding Cellular Automata and the related cryptographic terms. The list of hybrid CA rules and the reason to choose such rules are explained in section 3. In section 4, we briefly describe the model of our analysis. d -monomial test is introduced in section 5 and the main results of d -monomial tests are also presented in that section. We also draw certain observations from the experimental results in respect of constructing cryptographically suitable hybrid CA with respect to d -monomial tests in that section. Finally, the paper is concluded in section 6.

2 Preliminaries

In this section, we present the basic definitions of CA and also of the cryptographic properties.

2.1 Cellular Automata Related Definitions

Definition 1. *Cellular Automata: A cellular automaton is a finite array of cells. Each cell is a finite state machine $C = (Q, f)$ where Q is a finite set of states and f is a mapping $f : Q^n \rightarrow Q$. The mapping f , is called local transition function. n is the number of cells the local transition function depends on. On each iteration of the CA each cell updates itself with respective f .*

Adjacent cells of a cell are called the neighbourhood of CA. A 1-dimensional CA, whose rule depends on left and right neighbour and the cell itself is called a 3-neighbourhood CA. Similarly, if each cell depends on 2 left and 2 right neighbours and itself only, it is called 5-neighbourhood CA. A CA whose cells depend on 1 left and 2 right neighbouring cells is called a 4-neighbourhood right skew CA. A left skewed 4-neighbourhood CA can be defined similarly.

Definition 2. *Rule: The local transition function for a 3-neighbourhood CA cell can be expressed as follows:*

$$q_i(t+1) = f[q_i(t), q_{i+1}(t), q_{i-1}(t)]$$

where, f denotes the local transition function realized with a combinational logic, and is known as a rule of CA [9]. Here, $q_i(t)$ represents the value of the i^{th} cell after t iterations. The decimal value of the truth table of the local transition function is defined as the rule number of the cellular automaton.

For one dimensional 3-neighbourhood CA the definitions of some rules are given below:

Rule 30: $f = q_{i-1}(t) \oplus (q_{i+1}(t) + q_i(t))$, where $+$ is the Boolean 'OR' operator and \oplus is the Boolean 'XOR' operator.

Rule 60: $f = q_{i-1}(t) \oplus q_i(t)$.

Rule 90: $f = q_{i-1}(t) \oplus q_{i+1}(t)$.

A CA whose local transition function is same across the cells is called *uniform CA*. A CA whose local transition function is *not* same for all the cells is a *hybrid CA*. Hybrid CA may be constructed by choosing different linear rules or by choosing different linear and nonlinear rules over the automaton. A CA whose first and last cells are connected to 0 is called *null-boundary CA*.

A CA whose local transition function consists of only 'XOR' operator is called a *linear CA*. A CA whose at least one local transition function consists of 'AND'/'OR' in addition to 'XOR' is *nonlinear CA*. For example, rule, $f = q_{i-1}(t) \oplus q_{i+1}(t)$ employed in each cell is a linear CA and $f = q_{i-1}(t).q_{i+1}(t)$ employed in each cell is a nonlinear CA, where, $q_{i-1}(t)$ and $q_{i+1}(t)$ denotes left and right neighbours of the i^{th} cell at t^{th} instance of time. A uniform CA each of whose transition function is, $f = q_{i-1}(t) \oplus (q_{i+1}(t) + q_i(t))$ is a *rule 30 uniform CA*.

Any CA can be utilized to generate sequences by first selecting a *seed* and then updating each cell according to its transition function. State values from the middle cell of the cell array are output to represent generation of sequences. This sequence can be tested for pseudorandomness.

2.2 Cryptographic Terms and Primitives

We now present definitions of related cryptographic terms and properties used in this paper.

Definition 3. *Pseudorandom Sequence:* An algorithmic sequence is pseudorandom if it cannot be distinguished from a truly random sequence by any efficient (polynomial time) probabilistic procedure or circuit.

A variable or its negation (x or \bar{x}) is called a literal. Any number of 'AND'-ed literals is called a *conjunction*. For example, $x.y.\bar{z}$ is a *conjunction*.

Definition 4. *Algebraic Normal Form:* Any Boolean function can be expressed as XOR of conjunctions and a Boolean constant, True or False. This form of the Boolean function is called its Algebraic Normal Form (ANF).

Every Boolean function can be expressed in ANF. As an example, $f(x_1, x_2, x_3) = (x_1 \oplus x_2).(x_2 \oplus x_3)$ is not in ANF. Its ANF representation is, $f(x_1, x_2, x_3) = x_1.x_2 \oplus x_1.x_3 \oplus x_2 \oplus x_2.x_3$.

Definition 5. *Algebraic Degree: The maximum number of literals in any conjunction of ANF of a Boolean function is called its degree. Ciphers expressible or conceivable as a Boolean function have algebraic degree which is the same as the degree of the ANF of the Boolean function.*

Thus, $f(x_1, x_2) = x_1 \oplus x_2 \oplus x_1.x_2$ has algebraic degree 2.

Table 1. ANF of used 3 and 4-neighbourhood Rules

Type	Rule #	Nbd	ANF
Linear	15	3	\bar{x}_1
	60	3	$x_1 \oplus x_2$
	90	3	$x_1 \oplus x_3$
	150	3	$x_1 \oplus x_2 \oplus x_3$
	240	3	x_1
Non-linear	30	3	$(x_2.x_3) \oplus x_1 \oplus x_2 \oplus x_3$
	37	3	$x_1 \oplus x_2 \oplus (x_1.x_2.x_3) \oplus 1$
	45	3	$x_1 \oplus x_3 \oplus (x_2.x_3) \oplus 1$
	75	4	$x_1 \oplus x_2 \oplus x_2 \oplus (x_1.x_2) \oplus (x_1.x_3) \oplus (x_3.x_4) \oplus (x_1.x_3.x_4)$
	86	4	$x_2 \oplus x_3 \oplus x_4 \oplus (x_1.x_2) \oplus (x_1.x_3) \oplus (x_1.x_4) \oplus (x_2.x_3) \oplus (x_1.x_2.x_3)$
	91	3	$x_2 \oplus (x_1.x_2) \oplus (x_1.x_3) \oplus (x_2.x_3) \oplus (x_1.x_2.x_3) \oplus 1$
	120	3	$x_1 \oplus (x_2.x_3)$
	180	3	$x_1 \oplus x_2 \oplus (x_2.x_3)$
210	3	$x_1 \oplus x_3 \oplus (x_2.x_3)$	

3 Choice of CA Rules for Cryptographic Applications

Existing literature shows that only linear rules 90 and 150 and nonlinear rule 30 have been explored for cryptographic applications. Till date, uniform nonlinear and hybrid nonlinear rules have not been studied much.

We have considered 3 and 4-neighbourhood uniform nonlinear CA configurations. Combination of rule 30 and some other linear and nonlinear 3-neighbourhood rules have also been explored. The hybrid CA configurations are constructed by using the rules given in table 1.

The objective of the proposed construction is that the linear rules help to reduce the correlation, while the nonlinear rule provides required nonlinearity. The effect becomes prominent after a few initial iterations required to *mix* both these types of rules. The reason rule 30 is taken for the hybrid rulesets is that, it is nonlinear and it has good pseudorandom characteristics [10]. Though, rule 30 has good pseudorandom characteristics and it is a balanced rule, it has a strong correlation, namely, the probability, $Pr[x_i(t+1) = 1 \oplus x_{i-1}(t)] = \frac{3}{4}$, where $x_i(t)$ is the state of the i^{th} cell of the CA at t^{th} instance of time. The above property led to its cryptanalysis described by Willi Meier and Othmar Staffelbach [6].

We provide an analytical argument in favour of the fact that the introduction of linear rules reduce the correlation.

Consider two CA configurations, (i) Uniform rule 30 CA, C_1 ; (ii) Hybrid CA having alternate rules 30 and 60, C_2 . The input to both the CA are denoted by an array, $x_i, 0 \leq i \leq n$, where n is the length of the CA. Now,

1. In case of C_1 , $Pr[x_i(t+1) = 1 \oplus x_{i-1}(t)] = \frac{3}{4}$, for all time instances t . Hence, $Pr[x_i(t+2) = x_{i-2}(t)] = \frac{9}{16}$.
2. In case of C_2 , due to introduction of rule 60, i.e., $x_i(t+1) = x_{i-1}(t) \oplus x_i(t)$, we have, $Pr[x_i(t+1) = x_{i-1}(t)] = \frac{1}{2}$. Hence, there is no bias in predicting the output of the corresponding cell. This effect of unpredictability propagates to the nonlinear rule 30 in subsequent iterations. Therefore, $Pr[x_i(t+2) = x_{i-2}(t)] = \frac{3}{4} \times \frac{1}{2} = \frac{3}{8}$. Thus, the correlation gets reduced with each iteration. This justifies the construction of hybrid CA by alternating nonlinear and linear rules.

Evidently introducing more linear rules with nonlinear rules reduce correlations faster. For this reason we have considered a series of rule-30 based hybrid CA. In other words, the combination of rules is made in expectation of obtaining pseudorandom characteristics of rule 30 without the weakness of correlation.

For the experiment we have taken the following hybrid CA rulesets:

1. *Ruleset 1*: Rules 30 and 60 spaced alternately over a 3-neighbourhood CA.
2. *Ruleset 2*: Rules 30, 60 and 90 spaced alternately over a 3-neighbourhood CA.
3. *Ruleset 3*: Rules 30, 60, 90 and 120 spaced alternatively over a 3-neighbourhood CA.
4. *Ruleset 4*: Rules 30, 60, 90, 120 and 150 spaced alternatively over a 3-neighbourhood CA.
5. *Ruleset 5*: Rules 30, 60, 90, 120, 150, 180, 210, 240 spaced alternatively over a 3-neighbourhood CA.
6. *Ruleset 6*: Rules 30, 60, 90, 120, 150, 180, 210, 240, 15, 45 spaced alternatively over a 3-neighbourhood CA.

4 Functional Model of CA for Testing Crypto-Properties

For the experiment, we have taken an $n + 1$ -cell *null-boundary CA* (figure [II](#)). Here, without loss of generality, n is assumed to be odd. Each cell of the CA is assumed to have an unknown value, $x_i, 0 \leq i \leq n$ at the beginning. Boolean rules are set into the CA cells according to the CA configuration needed. Thus, each cell's output is determined by a corresponding local transition function f_i . Collectively, the functions are represented as F . The output bits of the CA are denoted by, y_0, y_1, \dots, y_n . The middle cell's output ($y_{\frac{n+1}{2}}$) is analyzed. Here, $f_{\frac{n+1}{2}}$ is the local transition function of the $\frac{n+1}{2}^{th}$ cell and $f_{\frac{n+1}{2}}^t$ is defined recursively as follows:

$$f_{\frac{n+1}{2}}^{t+1} = f_{\frac{n+1}{2}}(f_{\frac{n+1}{2}}^t)$$

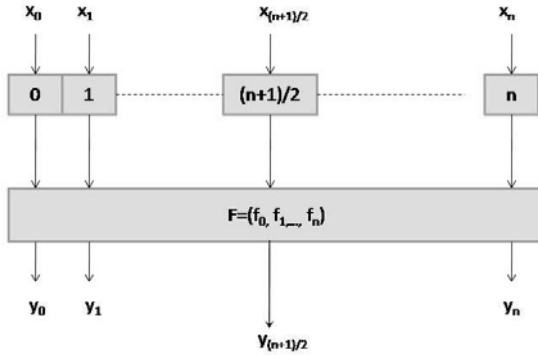


Fig. 1. Configurations of CA experimented

For that, we express the $\frac{n+1}{2}^{th}$ cell's output as a function of initial input unknowns, $x_i, 0 \leq i \leq n$. For a 3-neighbourhood CA,

$$y_{\frac{n+1}{2}} = f_{\frac{n+1}{2}}(x_{\frac{n+1}{2}-1}, x_{\frac{n+1}{2}}, x_{\frac{n+1}{2}+1})$$

and for a 4-neighbourhood right-skewed CA,

$$y_{\frac{n+1}{2}} = f_{\frac{n+1}{2}}(x_{\frac{n+1}{2}-1}, x_{\frac{n+1}{2}}, x_{\frac{n+1}{2}+1}, x_{\frac{n+1}{2}+2})$$

This process is iterated for multiple clock cycles, for 3 and 4-neighbourhood CA,

$$y_{\frac{n+1}{2}}^t = f_{\frac{n+1}{2}}^t(x_{\frac{n+1}{2}-t}, \dots, x_{\frac{n+1}{2}}, \dots, x_{\frac{n+1}{2}+t})$$

$$y_{\frac{n+1}{2}}^t = f_{\frac{n+1}{2}}^t(x_{\frac{n+1}{2}-t}, \dots, x_{\frac{n+1}{2}}, \dots, x_{\frac{n+1}{2}+2t})$$

Thus, it is clear that at t^{th} iteration, for a 3-neighbourhood CA, the output bit is a function of $2t + 1$ bits and for a 4-neighbourhood CA it is a function of $3t + 1$ bits. Beyond 3^{rd} iteration, the Boolean function acts upon 10 more variables and hence becomes unwieldy to analyze. In this paper, we have listed results of first 3 iterations only. We have chosen the $\frac{n+1}{2}^{th}$ cell for our analysis because it will be least effected by the boundary null values and more affected by the neighbouring cells and thus better characterize the rule of the CA. However, in case of hybrid configurations, we have analyzed output of all nonuniform middle cells and have selected the best rule as the output.

Historically, researchers have studied balancedness, nonlinearity, resiliency and algebraic degree [7], [8] to explore CA as a crypto-primitive. Our emphasis is on a new cryptographic test called d -monomial test.

5 *d*-Monomial Test

d-Monomial test is a statistical test for pseudorandomness introduced independently in [1] and [3]. It investigates the Boolean function representation of each output bit in terms of input bits. If a Boolean function of *n* Boolean variables is a good pseudorandom sequence generator, then it will have $\frac{1}{2} \binom{n}{d}$ *d*-degree monomials. The distribution is *binomial*. A χ^2 test with one degree of freedom is applied to count to measure how *unbiased* the count is. A deviation will indicate non-randomness. For example, consider the function $f(x_1, x_2, x_3) = x_1 \oplus x_2$, it has 2, 1-degree monomials and 0, 2 degree monomial. The ideal number of 1, 2 and 3 degree monomials would be $\frac{1}{2} \binom{3}{1} = 1.5$, $\frac{1}{2} \binom{3}{2} = 1.5$ and $\frac{1}{2} \binom{3}{3} = 0.5$. It turns out that it has 2, 1-degree monomials more and 1 2-degree monomial less, hence it is expected to be non-pseudorandom. On the other hand, $f(x_1, x_2, x_3) = x_1 \oplus x_2 \cdot x_3$ is expected to be a good pseudorandom generator.

In spite of its simplicity, this test gained huge appreciation in cryptography community. It proved to be a good tool in analyzing the degree of pseudorandomness of cryptographic systems. To the best of our knowledge, *d*-monomial test has not been applied to CA configurations previously. We explore different CA configurations under this test.

The *d*-monomial test can be considered a stronger form of pseudorandomness test than is captured by the cryptographic properties, balancedness, nonlinearity, resiliency and algebraic degree. Not much work has been done on constructing Boolean functions which satisfy *d*-monomial test. On the other hand, lot of work exists for making good balanced, nonlinear, resilient and high algebraic degree Boolean functions.

Note that, since *d*-monomial test does not output a single value, it is difficult to compare *d*-monomial characteristics of two Boolean functions. We have given preference to Boolean functions having ideal values in higher degree over ideal values in lower degree. This is justified as cryptanalysis of Boolean functions having higher degree terms is harder than Boolean functions having lower degree terms.

Example of Calculation of Ideal d-Monomial Value: Let, number of variables in the Boolean function be, *n* = 5 and let *d* = 4; then, ideal number of 4-degree terms will be, $\frac{1}{2} \binom{n}{d} = \frac{1}{2} \binom{5}{4} = 2.5$. We will approximate it to 2.

5.1 *d*-Monomial Test of Uniform CA

The experiment is done using *Mathematica 7.0 Student Edition*. Each *ith* cell of the CA is assumed to be initialized with an unknown Boolean value, *x_i*. The Boolean rules of the CA are simulated and value of each cell is updated per iteration. For example, if the CA operates itself uniformly with rule 30, the *ith* cell will have value $x_{i-1} \oplus (x_{i+1} + x_i)$ after the first iteration of operation, where, \oplus stands for Boolean 'XOR' operator and + stands for 'OR' operator.

3-neighbourhood CA: Among uniform rules, rule 37 and rule 91 have the best *d*-monomial characteristics. Table [2] lists values of number of *nth* degree

Table 2. Comparison of d -monomial characteristics of rule 30, 37 and 91

	Number of n^{th} degree terms						
Rules	1	2	3	4	5	6	7
Ideal	1,2,2	1,5,10	0,5,52	0,2,52	0,0,10	0,0,3	0,0,0
30	3,3,6	3,5,11	1,2,8	0,0,4	0,0,1	0,0,0	0,0,0
37	2,3,5	0,8,13	1,4,21	0,0,26	0,1,8	0,0,1	0,0,1
91	1,2,3	3,7,8	1,6,14	0,5,12	0,1,11	0,0,6	0,0,1

Table 3. Comparison of d -monomial Characteristics of 4-neighbourhood CA

	Number of n^{th} degree terms								
Rules	1	2	3	4	5	6	7	8	9
Ideal	2,3,5	3,10,22	2,17,30	0,17,210	0,10,126	0,3,210	0,0,30	0,0,22	0,0,5
75	0,0,5	0,0,13	2,6,21	1,3,26	0,2,10	0,2,8	0,0,8	0,0,2	0,0,6
86	0,0,0	0,0,0	2,0,0	1,5,0	0,6,3	0,3,14	0,4,15	0,0,8	0,0,2

terms in the generated Boolean functions over the three iterations for rules 30, 37 and 91. According to the table, rule 30 performs worse than rule 37 and 91. It fails in generating some higher degree terms (degree 6 and 7 for example). It also generates less number of 3, 4 and 5-degree terms than rule 37 and 91. It generates close to ideal number of 2-degree terms compared to rule 37 and 91. But as already mentioned, ideal number of lower degree terms is not as important as ideal number of higher degree terms in view of cryptanalysis. Rule 91 performs better than rule 37 in higher degree terms and should be given preference. The growth rate of number of n^{th} degree terms is quite fast for both rules 37 and 91. Note that, even this two CA are far distant from the ideal d -monomial characteristics.

4-neighbourhood CA: We look at their d -monomial test results of rule 75 and rule 86 in table 3. This two rules outperform the other rules experimented. Both rules, 75 and 86 have many higher degree terms and few or no lower degree terms. However, the number of higher degree terms is not ideal too. Between rules 75 and 86 we should choose rule 75 as it generates closer to ideal number of terms of each degree than rule 86. Again note that, even the best rules, rule 75 and rule 86 are far distant from the ideal d -monomial characteristics.

5.2 d -Monomial Test of Hybrid CA

d -monomial test results of the 6 hybrid CA rulesets are given in the table 4. The table above shows that, rulesets 5 and 6 are better rulesets than all other hybrid configurations, as they generate closer to ideal number of terms of almost all n^{th} degree terms than the other rulesets. Rules 37 and 91 have better d -monomial characteristics but the generated functions have low resiliency and are unbalanced compared to the hybrid counterparts.

Table 4. d -Monomial Characteristics of Hybrid CA

Rules	Number of n^{th} degree terms			
	1	2	3	4
Ideal	1,2,3	1,5,10	0,5,52	0,2,52
Ruleset 1	3,3,5	1,3,3	0,0,2	0,0,0
Ruleset 2	3,3,2	1,3,3	0,0,1	0,0,0
Ruleset 3	3,2,4	1,3,5	0,1,3	0,0,0
Ruleset 4	3,2,4	1,3,7	0,1,7	0,0,2
Ruleset 5	3,2,4	1,3,5	0,2,6	0,0,3
Ruleset 6	3,2,4	1,3,5	0,2,6	0,0,3

d -monomial characteristic is an important metric in finding which rules should be combined in a hybrid CA. As an example, let us form a CA consisting of rules 30 and 37 spaced alternatively. The d -monomial characteristics of the CA is given in table 5 along with characteristics of rule 30 and 37. We have seen that, rule 30 and rule 37 have good d -monomial characteristics. But, note that, the new CA performs even better than both the rules in higher degree terms (degree 5 onwards). At higher degree terms its d -monomial values are very close to ideal. In the middle and lower degree terms also the d -monomial values are good, though rule 37 has better values in this region.

Both rule 30 CA and rule 37 CA are heavy on higher degree terms. Their combination is expected to have more number of higher degree terms. Likewise, linear rules can not add terms of more than degree 1 to the generated Boolean function. But combining linear rules with higher algebraic degree rules, we will be able to add missing degree 1, 2 and other lower degree terms in the generated Boolean functions, which will perform better in d -monomial test. The above observation is important in hybrid CA constructions. This may be a way of reaching ideal d -monomial characteristics. We refer to this process as d -monomial characteristics addition. However, no direct relationship in d -monomial values of component rules and the hybrid rule can be inferred from the result (table 5). But, understanding the behaviour of this process is crucial in design of cryptographically suitable CA or good Boolean function generator.

Table 5. d -Monomial Characteristics Addition of Hybrid CA

Rules	Number of n^{th} degree terms						
	1	2	3	4	5	6	7
Ideal	1,2,2	1,5,10	0,5,52	0,2,52	0,0,10	0,0,3	0,0,0
30	3,3,6	3,5,11	1,2,8	0,0,4	0,0,1	0,0,0	0,0,0
37	2,3,5	0,8,13	1,4,21	0,0,26	0,1,8	0,0,1	0,0,1
(30, 37)	2,4,2	0,6,9	3,7,13	0,2,19	0,0,10	0,0,4	0,0,1

6 Conclusion

We have presented experimental results of d -monomial test of different configurations of uniform and hybrid CA. We have seen that it is possible to construct hybrid CA that can provide fair d -monomial characteristics and at the same time can be resilient, balanced and nonlinear. It is possible to improve d -monomial characteristics of CA rules by combining rules into a hybrid CA. We referred to this process as *d-monomial characteristics addition*. This property can be employed to form cryptographically suitable CA.

References

1. Filiol, E.: A new statistical testing for symmetric ciphers and hash functions. In: Deng, R.H., Qing, S., Bao, F., Zhou, J. (eds.) ICICS 2002. LNCS, vol. 2513, pp. 342–353. Springer, Heidelberg (2002)
2. Johansson, T., Englund, H., Turan, M.S.: A framework for chosen iv statistical analysis of stream ciphers. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 268–281. Springer, Heidelberg (2007)
3. Juhani, M., Saarinen, O.: Chosen-iv statistical attacks on e-stream stream ciphers. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/013, pp. 5–19 (2006)
4. Murphy, S., Paterson, K.G., Blackburn, S.R.: Theory and applications of cellular automata in cryptography. IEEE Transactions on Computers, 46(5) (1997)
5. Koc, C.K., Apohan, A.M.: Inversion of cellular automata iterations. IEE Proceedings of Computers and Digital Techniques 144(5), 279–284 (1997)
6. Meier, W., Staffelbach, O.: Analysis of pseudo random sequences generated by cellular automata. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 186–199. Springer, Heidelberg (1991)
7. Martin, B., Sole, P.: Pseudo-random sequences generated by cellular automata. In: International Conference on Relations, Orders and Graphs: Interactions with Computer Science (2008)
8. Martin, B., Sole, P., Lacharme, P.: Pseudo-random sequences, boolean functions and cellular automata. Boolean Functions: Cryptography and Applications (2007)
9. Roy Chowdhury, D., Nandi, S., Chattopadhyay, S., Pal Chaudhuri, P.: Ca and its applications: A brief survey. Additive Cellular Automata - Theory and Applications (1997)
10. Wolfram, S.: Cryptography with cellular automata. CRYPTO: Proceedings of Crypto (1985)
11. Wolfram, S.: Random sequence generation by cellular automata. Advances in Applied Mathematics 7, 123–169 (1986)

Programmable Cellular Automata (PCA) Based Advanced Encryption Standard (AES) Hardware Architecture

Nirmalya S. Maiti, Soumyabrata Ghosh,
Biplab K. Shikdar², and P. Pal Chaudhuri

¹ Cellular Automata Research Lab (CARL),

Alumnus Software Limited, Sector - V, Kolkata, India 700091

² Bengal Engineering & Science University, Shibpur, Howrah, India 711103

maitinirmalya@gmail.com

Abstract. This paper reports an AES (Advanced Encryption Standard) hardware architecture based on Programmable Cellular Automata (PCA) operated with a program. Verilog code has been designed for PCA and associated hardware modules to realize the AES functions. The PCA replaces the irregular logic circuit necessary for hardwired implementation of AES. The design has been simulated on Xilinx platform using ISE simulator.

Index Term: AES, PCA.

1 Introduction

Rijndael algorithm [5] was selected as the Advanced Encryption Standard (AES). It is a symmetric byte-oriented iterated block cipher scheme. High popularity of AES algorithm stems from its inherent strength in respect of security, performance, and efficiency. In view of its commercial interest, the hardwired implementation of AES has received considerable attention in recent years [2-4]. Simple, regular, modular, cascadable, and local neighborhood structure of a PCA suits ideally for VLSI design of complex AES functions like multiplicative inverse for Sub-Byte, MixColumn etc. and their inverse operations.

2 Programmable Cellular Automata (PCA) and the Program Structure

A CA $\langle R_0 R_1 \dots R_i \dots R_{n-1} \rangle$ evolves in discrete time steps with a specific rule R_i employed on the i^{th} cell ($i = 0, 1, \dots, (n-1)$). The generalized hardware structure of a Programmable CA (PCA) cell (introduced in [1]) is shown in Fig 1(a). A PCA cell allows implementation of any one of 256 three neighborhood CA rules. It employs an 8 to 1 Multiplexer that accepts 8 binary bits of a CA rule. The output of the Multiplexer is fed as the input to the cell memory element.

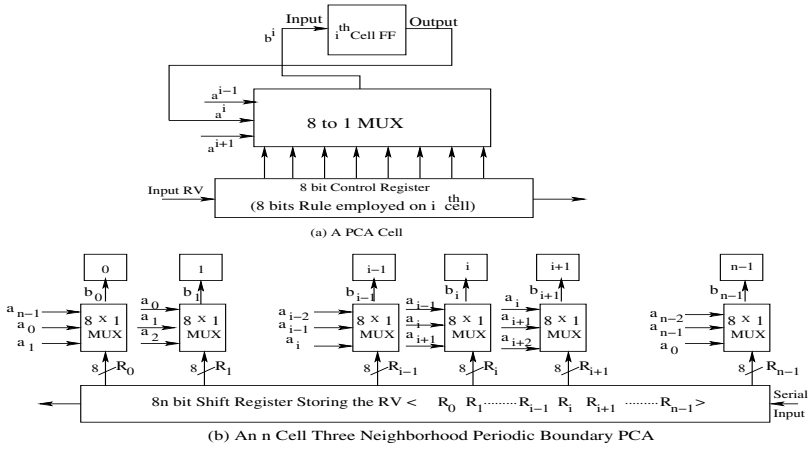


Fig. 1. A PCA cell and Programmable CA (PCA) Structure

Multiplexer control bits are derived from $(i - 1)^{th}$, i^{th} , and $(i + 1)^{th}$ cell outputs. Fig 1(b) shows an n cell periodic boundary PCA. In addition to n cells, the PCA has an $8 * n$ bit register to store the Rule Vector (RV) of the PCA.

An n cell PCA initialized with the seed $S_0 = \langle a_0 a_1 \dots a_i \dots a_{n-1} \rangle$ is operated with program instruction stored in memory (Fig 2). The program instruction, as shown in Fig 2(b), has three fields. An n cell PCA has $8n$ bits in Field 1 - eight bit per cell to represent the rule employed for a cell. In a particular program step, the PCA is first configured as a specific CA as per the Rule vector defined by Field 1. This is followed by operation of the CA for the number of time steps (t) specified in Field 2. The input S_0 gets transformed in discrete time steps of CA evolution. At the end of execution of a program step, the CA has the state S_{0+t} derived on running the CA for t time steps with S_0 as the seed. The Field 3 of program instruction consists of tag and control bits used for control of program flow, boundary etc.

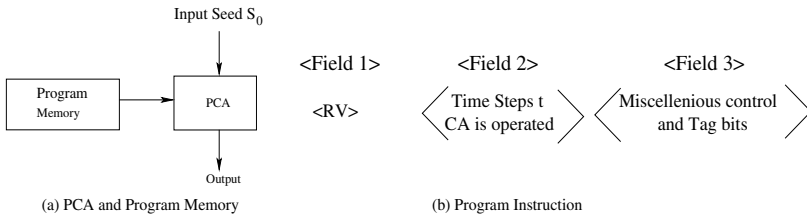


Fig. 2. PCA operated by program

3 PCA Based Processor to Realize AES

The generic PCA structure (of Fig 1) has been modified, as shown in Fig 3, to realize AES hardware architecture. Other hardware blocks used for the processor, as shown in Fig 4, are - Plain / Cypher Text Register (16 X 8), Key Register (256 X 8), X-OR Unit, 32 Bit Shift Register Unit, Round Counter, Column Counter, Row Counter. All bytes in the AES algorithm are interpreted as finite field elements in $GF(2^8)$. Multiplication operation is done over $GF(2^8)$ employing modulo of irreducible polynomial $x^8 + x^4 + x^3 + x + 1$ and generator polynomial as $(x + 1)$. Multiplication can be implemented with the 8 cell PCA shown in Fig 3.

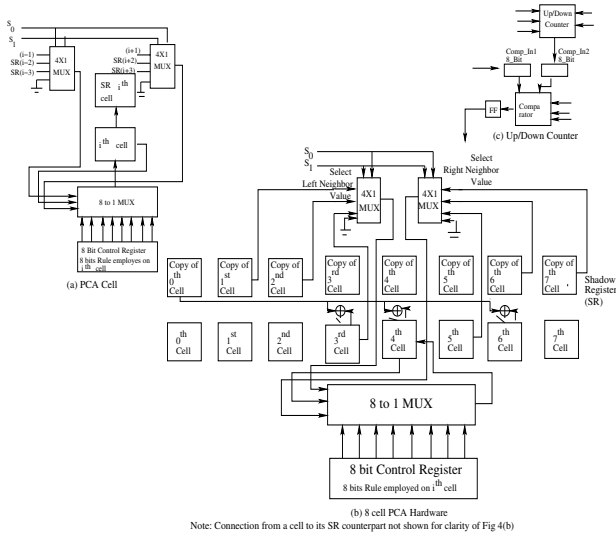


Fig. 3. Cell and PCA Structure for 8-bit processor

4 Realization of AES Algorithm with 8 Bit Processor

AES Encryption implements four different transformations - (i) Sub-Byte Transformation, (ii) Mix-Column Transformation, (iii) Shift Row Transformation, and (iv) Add Round Key. Key Expansion is another task to generate keys for different Rounds. Multiplicative Inverse, Affine transform, and Mix-Column transformation can be efficiently realized through PCA evolution.

The AES algorithm takes the Plain Key (K), and performs a Key Expansion routine to maintain the key scheduling. PCA program for each of these transformations has been developed. The 8 bit processor realizing AES has been simulated on Xilinx platform using ISE simulator for encryption and decryption of randomly generated plain text. Verilog code has been designed for PCA and associated hardware modules to realize AES hardware.

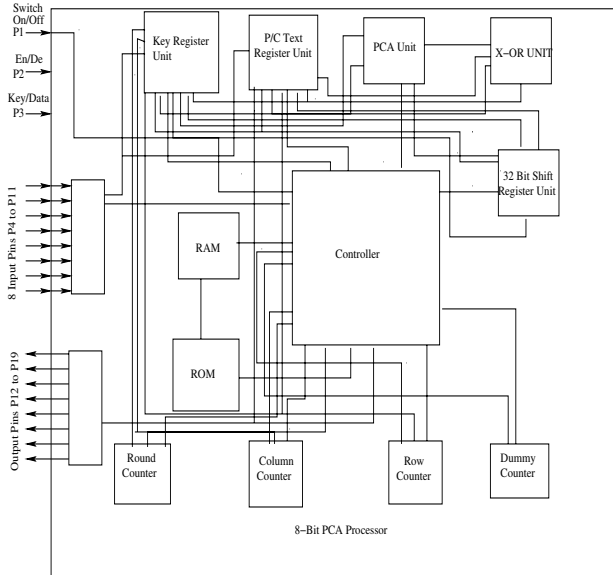


Fig. 4. 8 Bit Processor Realizing AES

Conclusion. The regular and modular structure of PCA operated with program instructions (Fig 2(b)) replaces the irregular logic employed for AES hardware reported in published literature.

References

1. Chaudhuri, P.P., Chowdhury, D.R., Nandi, S., Chatterjee, S.: Additive Cellular Automata, Theory and Applications, vol. 1. IEEE Computer Society Press, Los Alamitos (1997) ISBN-0-8186-7717-1
2. Gaj, K., Chodowicz, P.: Hardware Performance of the AES finalists - survey and analysis of results, http://ece.gmu.edu/crypto/AES_survey.pdf
3. Park, S.J.: Analysis of AES Hardware Implementations, <http://cs.ucsb.edu/~koc/cs290g/project/2003/park.pdf>
4. Alam, M., Ray, S., Mukhopadhyay, D., Ghosh, S., RowChowdhury, D., Sengupta, I.: An Area optimized Reconfigurable Encryptor for AES-Rijndael. In: DATE (2007)
5. Daemen, J., Rijmen, V.: The Design of Rijndael. Springer, Heidelberg (2002)

Exhaustive Evaluation of Radius 2 Toggle Rules for a Variable-Length Cryptographic Cellular Automata-Based Model

Gina M.B. Oliveira, Luiz G.A. Martins, Leonardo S. Alt,
and Giordano B. Ferreira

Faculdade de Computação, Universidade Federal de Uberlândia – UFU
Av. João Naves de Ávila, 2121- Campus Santa Mônica, Bloco B, sala 1B60
CEP: 38400-902 Uberlândia, MG, BRA
Tel.: +55 (34) 3239-4144
gina@facom.ufu.br

Abstract. A cellular automata (CA) model in cryptography is investigated. A previous work analyzed the usage of reverse algorithm for pre-image computation as an encryption method. The main conclusion was that the simple adoption of such method is not viable, since it does not have 100% of guarantee of pre-image existence. A new approach was proposed that uses extra bits when the pre-image computation is not possible. It is expected that in practice few failures happens and the ciphertext size will be close to the plaintext. Encryption always succeeds and the final length of the ciphertext is not fixed. We better investigate the secret key specification by using a more representative set formed by all radius 2 right-toggle rules, totalizing 65536 rules. An exhaustive analysis of this rule space has shown that using adequate specification the method has a good protection against differential cryptanalysis and a small increase in ciphertext length.

Keywords: Cellular Automata, cryptography, pre-image computation.

1 Introduction

Cellular automata (CA) are particularly well suited for cryptographic application and there are several previous studies in this topic [1,3,4,5,6,7,8,9]. Since CA rule is simple, local and discrete, it can be executed in easily-constructed massively-parallel hardware at fast speeds. Considering the reverse interaction of a cellular automaton, given a lattice in time t , a possible antecessor lattice is determined for the time $t - 1$. This process is also known as pre-image computation. In recent papers [8,9,10], the application of the reverse algorithm proposed by Wuensche and Lesser [11] as a cipher algorithm was investigated. Besides, static parameters were employed to specify CA rules as appropriate secret keys in [8] and [9]. The objective of the employment of such parameters was to find rules with 100% guarantee of pre-image existence for any possible lattice. Using

a spatial entropy measure to evaluate the ciphering quality, the main conclusion in [8] is that the simple adoption of the reverse algorithm is not possible because the only rules with 100% guarantee of pre-image existence are not appropriate for ciphering because they do not exhibit a chaotic dynamics. A new approach has been emerged from this previous study [8]: it alternates the original reverse algorithm and the variation that uses extra bits, when the pre-image computation fails. This variation is similar to pre-image computation adopted in Gutowitz model. Although this approach needs to add bits to the ciphertext when a failure occurs, it is expected that in practice few failures happen and the ciphertext length will be equal or close to the plaintext. CA rules used as secret keys must be properly specified to obtain this low probability of failure occurrence during pre-image computation. It was shown in [8] that the joint use of symmetry (S) and Z parameters – and components Z_{left} and Z_{right} – could lead us to a good specification of rules which have low probability to fail in pre-image computation. This analysis was performed using small samples of radius 2 and 3.

In the present work, we better investigate the secret key specification. First, we use a more representative rule set formed by all radius 2 right-toggle rules, totalizing 65536 rules. These rules represent 50% of the possible secret keys in radius 2 space, being that the other 50% are the all radius 2 left-toggle rules, which are dynamically equivalent to the set analyzed. Based on an exhaustive analysis of this rule space we analyze the effects of using: (i) the complete set of rules; (ii) the restricted rule set defined by the specification proposed in [8]; (iii) the restricted rule set defined by the specification proposed in [9]. The main conclusion is that there are a lot of undesirable behavior rules in the complete set – considering a cryptographic purpose – that must be avoided as secret keys. Thus, in a subsequent phase, we employed an analysis based on several CA static parameters, trying to capture the pattern associated to these underperforming rules. Using them, we were able to find a good specification of rules to be used as valid secret keys. This specification had shown to be good to filter the set of radius 2 rules and to elaborate new radius 3 rules.

2 Cellular Automata Parameters

A cellular automaton consists of a lattice of cells and a transition rule. Each cell presents in each time t one of k distinct states. A cell is updated in discrete time steps and its new state depends on the states of the $2R + 1$ neighborhood cells, where R is the CA radius. In the case of a deterministic one-dimensional CA, the state a_i^{t+1} of the cell i in time $t + 1$ is determined by the transition rule τ :

$$a_i^{t+1} = \tau[a_{i-R}^t, \dots, a_i^t, \dots, a_{i+R}^t]. \quad (1)$$

The dynamics of a CA is associated with its transition rule. In order to help forecast CA dynamical behavior, several parameters have been proposed in the literature [12,13], as: (i) Z derived from reverse algorithm, composed by Z_{left} and Z_{right} [14]; (ii) S is the symmetry level of a rule transition (b_1, b_2, \dots, b_k)

given by the number of pair of bits b_i and b_{k-i+1} ($i = 1, \dots, k/2$) that have the same value [8];(iii) Absolute activity (AA) quantifies how much change is entailed by the rule, in the state of the centre cell, in relation to the current state of the centre cell, and the states of the pair of cells which are equally apart from the centre [13].

3 Previous CA-Based Cryptographic Models

Wolfram (1986) was the first to suggest the use of CA in cryptography [1]. Several studies on this topic have been accomplished [3,4,5,6,7,8,9,10,11]. Gutowitz proposed a cryptographic model based on backward evolution of irreversible CA [6]. A toggle rule with radius R is used as the secret key in his model. A CA toggle rule is sensible in respect to a specific neighborhood cell - any modification of the state on this cell necessarily provokes a modification on the new state of the central cell. A pre-image of an arbitrary lattice of size N is calculated adding R extra bits in each side of the lattice. Considering a rule transition with right-toggle property, it guarantees that the entire $N + 2R$ pre-image cells can be obtained, step-by-step, from the leftmost side to the right, in a deterministic way [6]. Plaintext is the initial lattice and P pre-images are calculated. The ciphertext is given by the last pre-image obtained. As $2R$ bits are added to each pre-image calculated, the size of the final lattice is given by $N + 2RP$. Such non-negligible increment is pointed as the major flaw in Gutowitz's model.

An algorithm known as reverse algorithm was proposed in [11] for a generic pre-image computation. Before starting the computation, R cells are added to each side of the lattice corresponding to the pre-image, in a similar way to Gutowitz's pre-image computation. However, using a periodic boundary CA, pre-image computation is concluded verifying if the initial bits can be equal to the final $2R$ rightmost ones. If so, the extra bits are discarded returning the pre-image to the same size of the original lattice. This algorithm finds all the possible pre-images for any arbitrary periodic boundary lattice. Reverse algorithm was evaluated as an encryption method in [8] and [9]. However, its application in ciphering has the disadvantage that there is no guarantee of pre-image existence for any given lattice and any given rule transition. Therefore, the major challenge to apply reverse algorithm as a viable cipher method was to guarantee the existence of at least one pre-image for any possible lattice. The first attempt to solve this problem was to use Z [14] in rule specification [8,9].

An analysis of the secret keys was performed in [8]. Parameters that have presented more dependence to the 100% of pre-image existence were the components of Z - Z_{right} and Z_{left} - and the symmetry S . It was observed that one component of Z must be equal to 1 and the other one must be different of 1. Moreover, S equal to 1 also must to be avoided. The best initial experiments performed in [8] have applied rules with low values both for symmetry and Z_{left}/Z_{right} balance: $0 < S < 0.25$ and $0 < Z_{left} < 0.25$ and $Z_{right} = 1$. Using this specification the rules could find ten consecutive pre-images for almost all evaluated plaintexts: 99.9997% of 3×10^6 of 512 bits-lattices. However, using a

entropy measure to evaluate the quality of ciphering performed by this previous rule set, a trade-off was recognized when specifying a rule transition to be used with the reverse algorithm: if the rule is perfect in respect to the existence of pre-images, it does not have a chaotic behavior; if the rule is perfect in respect to the chaoticity, it cannot be able to calculate the pre-image for a large range of possible plaintexts. The best experiments performed have applied rules with an intermediate level both for symmetry and Z_{left}/Z_{right} balance: $0.25 < S < 0.5$ and $0.25 < Z_{left} < 0.5$ and $Z_{right} = 1$. Besides, the average of the mean entropy obtained for all the rules was high indicating that they exhibit a chaotic behavior. An important observation is that although it was possible to specify rules with a high probability to find at least one pre-image for any lattice and with a good perturbation spread, even the better rules evaluated can fail when the pre-image computation is applied. The major conclusion in [8] is that the simple adoption of the reverse algorithm is not viable because the possible rules with 100% guarantee of pre-image existence are not appropriate for ciphering; they only shift the lattice. An alternative approach has emerged in [8]: a method based on reverse algorithm adopting a contour procedure to apply when pre-image computation fails. The contour procedure guarantees the possibility to cipher any plaintext. In the present work, we discuss this method.

In a certain sense, the method proposed in [9] is very similar to the initial method proposed in [8], even so the methods have been proposed in an independent way. However, no treatment was addressed in [9] to failure occurrences when computing pre-images - an important point to discern the works in [8] and [9]. The simple secret key discarding in the case of a failure as suggested in [9] cannot be adopted in a communication system.

4 Variable Length Encryption Method

Since the main conclusion of the analysis in [8] is that the simple adoption of the reverse algorithm is not possible, an alternative method is investigated here. It is based on reverse algorithm adopting an alternative procedure to apply when the pre-image computation fails [8]. It is expected that with an appropriate key specification there is a low probability to this failure occurrence. The alternative procedure adds extra bits only when the pre-image is not possible to calculate. Therefore, it is expected to rarely use this procedure but it guarantees the possibility to cipher any plaintext. For practical reasons related to encryption speed, it can be better to limit the method to operate with only toggle rules. The method works as it alternates rounds of pre-image computation performed by reverse algorithm (a variation of Gutowitz's model for periodic conditions) with few or none steps of pre-image computation performed by Gutowitz's model. Ciphering is made by computing P consecutive pre-images starting from a lattice of size N corresponding to the plaintext. The secret key is a radius- R CA rule τ generated with an appropriate specification based CA static parameters.

Suppose that it started to calculate pre-images using reverse algorithm and the secret key τ and it fails in the K -th pre-image such that $K \leq P$. In such

situation the ciphering process uses the modified reverse algorithm with extra bits to calculate the K -th pre-image. Thus, the K -th pre-image will have $N + 2R$ cells. Ciphering returns again using the original reverse algorithm to calculate the remaining pre-images. If all the subsequent pre-images computation succeeds the final ciphertext will have a size of $N + 2R$. If the pre-image computation fails again, the ciphering process changes and adds $2R$ more bits to the lattice. If the process fails in F pre-images ($F \leq P$) the final lattice will have $N + 2FR$. Starting from a lattice of N cells, the size of the ciphertext after P pre-images computation is given by $N \leq \text{ciphertext size} \leq N + 2PR$. Therefore, it is a variable-length encryption model, named *VLE*. However, we expected that in practice the ciphertext size will be next to N due to the characteristics of the rule used as secret key. It is important to note that the ciphertext obtained using Gutowitz's model will have exactly $2PR$ bits – the worst and improbable situation in the proposed method. By starting from the ciphertext the recipient needs to apply the transition rule τ forward by P steps and the final lattice will be the plaintext. He also needs to know in which pre-images failures happened to recover the original text. An improvement of the method in relation to the one proposed in [8] is the usage of a non-retroceding method in a case of failure. When the pre-image computation fail, the method changes to the computation using extra bits without trying to backtrack to find another pre-image with lower order.

5 Experiments

Using VLE we have the guarantee that ciphering is possible even if an unexpected Garden-of-Eden state occurs. However a short length ciphertext depends on the secret key specification. Some experiments were performed in [10] to analyze method's performance and to evaluate previous rules specification. Our expectative about the usage of the variable-length method had been confirmed [10]: it has a good quality of ciphering entropy and the ciphertext length is close to the original block size, specially using rules specified according to [8]. However, a lot of open questions remain since experiments in [10] were performed based on very limited samples of rules (500 rules of radius 2 and 3). Here, we perform a deeper investigation about an appropriate rule specification. A representative key set was chosen considering radius 2 rule transitions. Aiming to perform a more exhaustive analysis new experiments were conducted using the complete set of radius 2 right-toggle rules; all of them have $Z_{left} = 1$ and $0 \leq Z_{right} \leq 1$. As a radius 2 toggle rule is defined by only 16 bits since the other 16 bits are deterministically defined. This set is composed by 65536 (2^{16}) rules, being that all of them have $Z_{left} = 1$. These rules represent 50% of the possible keys in radius 2 space (restricted to use only toggle rules for faster encryption), being that the other 50% are the left-toggle rules ($Z_{left} = 1$), which are dynamically equivalent to the set analyzed.

We employed the environment implemented based on the VLE to cipher a hundred 256-bits plaintexts using each right-toggle rule of the complete set. In

previous works [8] and [10] the number of consecutive pre-images employed during ciphering was fixed ($P = 128$), in which radius 2 rules and 256-bits plaintexts were also used. In the present work, the number of consecutive pre-image steps was dynamically defined between 16 and 128, depending on the results obtained during ciphering. In that way, the ideal number of consecutive pre-images employed during ciphering was also experimentally investigated. Based on an exhaustive analysis of radius 2 right-toggle rule space we analyze the effects of using as secret keys the following sets of rules: (i) *Fullset* - the complete set of rules in which the unique restriction is the right-toggle property ($Z_{left} = 1$ is a consequence); (ii) *Subset_A* - the restricted rule set defined by the specification $0.25 < Z_{right} \leq 0.5$ and $S \neq 1$ as proposed in [8]; (iii) *Subset_B* - the restricted rule set defined by the specification proposed in [9]: $0.5 \leq Z_{right} < 1$. *Fullset* is formed by 65536 rules, *Subset_A* by only 21019 rules (32.1% of *Fullset* rules) and *Subset_B* by 52801 rules (80.6% of *Fullset* rules). Therefore, the first distinction between the two subsets is that in *Subset_A* the reduction is much more severe and the number of available secret keys is reduced to approximate $\frac{1}{3}$.

The objectives of this investigation are: (i) *Calculating ciphertexts final length*: applying VLE method, any rule with $Z_{left} = 1$ is able to complete the ciphering process starting from any initial lattice. However, the final length of the ciphertext can be between N and $N + 2PR$. We want to evaluate if the expected final length is in fact close to the best case and which is the behavior of each group in such evaluation. Table 1 presents set performances: the average length of the final lattice or ciphertext (L_{mean}) and the average number of failures occurred during ciphering process (F_{mean}). Each average result was computed considering the application of all rules to cipher 100 lattices of 256 bits; (ii) *Comparing ciphertexts generated by pairs of similar plaintexts*: cryptanalysis methods try to find the plaintext after getting the ciphertext without knowing the secret key. The differential cryptanalysis is based on the analysis of some pairs of ciphertexts generated after similar plaintexts. Although the origins of differential cryptanalysis are related to studies in how to break DES algorithm [16], Sen et al. (2001) used it to analyze their CA cryptosystem named CAC and they compared their results with the results obtained with DES and AES cryptosystems [15]. In this analysis, several pairs of plaintext (X, X') are used, which differ one of the other by a fixed small difference D . Each pair (X, X') is used to generate a pair of ciphertexts (Y, Y') which differ one of the other by a difference D' . D and D' are obtained by applying XOR operations between the pair members. For each pair (Y, Y'), the number of 1s in D' is counted, which corresponds to the number of different bits between Y and Y' . Finally, the standard deviation of this measure is calculated over all the analyzed pairs. As higher is the standard deviation in D' , as higher is the probability of the ciphertext to be broken by differential cryptanalysis. An algorithm with standard deviation below 10% is said to be protected against differential cryptanalysis [15]. Difference D between X and X' was fixed in only one bit in an arbitrary position and the value of D' was determined for each plaintext evaluated. D' was calculated to each pair (Y, Y') to obtain the standard deviation (σ) for each rule set. Besides,

difference D' was also used to compute a second measure related to ciphering quality. The goal of this measure is to verify if D' does not keep any pattern which eventually could help a cryptanalyst. Spatial entropy [8] was calculated on D' to evaluate the existence of some undesirable regularity on this difference. Entropy above 0.75 indicates a random difference enough to expect that ciphertexts Y and Y' do not maintain any similarity, even so they started from similar plaintexts. Entropy below 0.5 indicates a strong pattern in D' [8], that is, a low ciphering quality. Entropy between 0.5 and 0.75 had been considered fuzzy, since it cannot guarantee the existence of an ordered or random pattern. Therefore, if any cryptography method is applied to similar texts returning an average of spatial entropy (E_{mean}) above 0.75, it indicates that ciphering adds a high entropy during the process, a necessary characteristic in any encryption method. Table 1 shows σ_{mean} and E_{mean} . We also used E_{mean} to determine when stop pre-image computation (P). D' entropy was calculated in some P steps to determine in which one the ciphering will be stopped. D' entropy is first calculated in $P = 16$. If it is above 0.75 the pre-image computation stops and the 16th pre-image is the ciphertext. Otherwise, this process is repeated to $P = 32, 64$ and 128. If until $P = 128$ entropy does not meet 0.75, the ciphertext is the 128th pre-image.

A mean standard deviation (σ_{mean}) below 5% was obtained for all the sets of rules. Therefore, the proposed CA cryptographic model can be considered secure in relation to differential cryptanalysis. The mean number of faults (F_{mean}) during ciphering process is very low for all set - below 0.1 - which returns a mean ciphertext size very close to the original size 256 bits. $Subset_A$ returned the best values, indicating that this specification indeed reduce the final size as pointed in [8]. When considering the mean entropy of D' (E_{mean}), all sets returned high values, above 0.87, indicating that the rules are able to add a high entropy during ciphering. Therefore, considering only the mean values of each set analyzed, all of them returned good values on the measures analyzed and there was no need to reduce the set of keys.

However, as pointed in [8], the worst performing rules in $Fullset$ indicate the existence of secrete keys not appropriate for ciphering purpose. Such rules are highlighted in Table 1, in which only the 500 worst performing rules in each set are considered. The mean number of faults (F_{mean}) is above 5 in $Fullset$ and above 3 in $Subset_B$, indicating that these rules returning ciphertexts with size superior to 270 bits in average. Moreover, F_{mean} represents the mean value size for each rule considering all the 100 lattices used to test it. However, if we consider the worst result in such lattices, we can find ciphertexts with a considerable size: column F_{max} shows the mean of the maximum ciphertext size obtained considering the 500 worst rules in each set. This metric highlights the existence of secret keys in $Fullset$ and $Subset_B$ returning ciphertexts with size superior to 290 bits. This undesirable number of fails can also be recognized observing Figure 1a: the maximum ciphertext size observed for each rule are plotted, considering the 200 worst rules in such metric. Figure 1b plots mean values (considering 100 plaintexts) for these rules. We can observe that there

are more than 200 rules in *Fullset* and *Subset_B* returning ciphertext lengths between 280 and 740 bits in the worst case and more than 270 bits in average. There are only 17 rules in *Subset_A* that returns ciphertext length above 300 bits in the worst case and only 97 rules with mean value between 260 and 280 bits. Therefore, considering the final ciphertext length, rules of *Subset_A* presented much better performance both in mean values considering the entire rule set and in mean and maximum values considering the worst rules (Table 1 and Figure 1). Naturally, this better performance is a consequence of the low number of fails.

Table 1. Mean values obtained for all rules and for the 500 worst rules in each set

Set	Number of rules	All rules				500 worst rules			
		L_{mean}	F_{mean}	E_{mean}	$\sigma_{mean}(\%)$	F_{mean}	F_{max}	E_{mean}	E_{min}
Fullset	65536	256.38	0.095	0.879	3.66	5.851	13.448	0.170	0.036
<i>Subset_A</i>	21019	256.10	0.025	0.872	3.65	0.932	3.212	0.396	0.037
<i>Subset_B</i>	52801	256.32	0.079	0.886	3.65	3.482	8.520	0.768	0.323
<i>Subset_C</i>	48689	256.26	0.064	0.887	3.31	1.875	5.318	0.858	0.749

No significant difference is clear in mean entropy considering all rules in Table 1. Table 1 also presents the 500 worst performing rules considering entropy values. The mean entropy in D' (E_{mean}) is below 0.5 in *Fullset* and *Subset_A*, indicating that they do not perform an actual encryption of the plaintexts in average. E_{min} represents the worst entropy found for each rule considering all the 100 lattices. E_{min} is below 0.1 for *Fullset* and *Subset_A*. It indicates the existence of lattices that are not encrypted by some few rules. The most probable behavior of such CA rules is that they only shift the initial lattices, not performing an actual encryption of plaintexts. Although this behavior is a minor occurrence in the entire set of CA rules it cannot be allowed in a cryptosystem.

This undesirable low entropy can also be recognized observing Figure 1c, in which the minimum D' entropy observed for each rule are plotted, considering the 1000 worst rules in such metric. Figure 1d plots the mean values (considering the 100 plaintexts evaluated) for these same rules. Considering the entropy of D' , rules specified as *Subset_B* presented much better performance both in mean values considering the worst rules and in minimum values considering the worst rules (Table 1 and Figure 1). About 400 rules in *Subset_B* returned entropy below 0.7 for at least one pair Y and Y' . However, due the gravity of this situation – where no encryption is in fact performed – the desirable number of such underperforming rules is zero. Concluding our analysis: (i) There are a lot of undesirable behavior rules in *Fullset* - considering a cryptographic purpose - that must be avoided as secrete keys. Therefore the entire rule space formed by all radius 2 toggle rules (totalizing 130,816 rules including left-toggle and right-toggle and the existence of 256 left-and-right-toggle) cannot be applied as secret keys in VLE method. Approximate 6000 rules must be avoided (3500 due to low entropy and 2500 due to long ciphertext length). (ii) Specifications was

proposed in [8] and [9] trying to filter such undesirable behavior keys. However, their application is so effective. The specification in [8] was proposed with the major goal of reducing ciphertext lengths, what is really achieved. But, the reduction imposed in key space is high (approximate 66%) and it could not avoid a great number of low entropy rules. The specification in [9] was proposed with the major goal of avoiding a gradual lead-in and lead-out of structure in the space-time pattern. As a consequence of this limitation on Z_{right} range, a great number of low entropy rules are avoided. However, about 500 low entropy rules remain in this subset and there are a lot of rules returning undesirable long ciphertext. A reasonable reduction is imposed in key space using this specification: approximate 20%.

Aiming to better understand the relation between CA static parameters and underperforming rules, several parameters was used trying to identify a pattern to filter such rules, as Z_{left} , S and Absolute Activity (AA) [13]. A new parameter was also used: it is the spatial entropy associated to the 16 bits that define the 32-bits toggle rule (the rule core), named here as core entropy (CE). All these parameters were calculated for all the 65536 radius 2 rules. After several analyses we have found the following rule to remove the worst performance CA transition rules from the entire key space. Any CA toggle rule can be used as a secret key if attempt the conditions: $S \neq 1$ AND $AA \geq 0.46$ AND $CE > 0.65$ AND $Z_{right} \neq 1$.

We applied this rule in the *Fullset* to filter all rules no attending its conditions. The filtered set named *Subset_C* has 48689 right-toggle rules. We employed this rule set using the environment implemented based on VLE to cipher a hundred 256-bits plaintexts. Considering the 48689 remaining rules, Table 1 and Figure 1 show results obtained with *Subset_C* rules. These mean values are better than those obtained using the entire radius 2 set. Only 32 rules with inappropriate low entropy and 198 rules with inappropriate ciphertext length remains in this filtered set. The performance of *Subset_A* related to long ciphertexts is still better than *Subset_C*, but results of the latter subset approximates for the first. Considering the aspects of our analysis - reduction of the entire key space, rules returning low D' entropy and rules returning long ciphertexts - we can consider *Subset_C* as a trustful key space for VLE method. Besides, a mean standard deviation (σ_{mean}) below 5% was obtained. Therefore, the proposed CA cryptographic model can be considered secure in relation of differential cryptanalysis when using *Subset_C* rules.

In respect to the number of pre-images P used during ciphering, the average of the maximum number of P used for each radius 2 rule of the *Fullset* was of 33.10 pre-images (considering the 100 plaintexts of 256-bits analyzed). Considering only the 48689 rules of *Subset_C* this mean value falls to 26.16. Thus, we conclude that a fixed number of pre-images $P = 32$ could be enough to cipher 256-bits plaintexts although $P = 128$ was used in previous works [8] and [10].

The performance in radius 2 space was satisfactory and the generalization ability of the filter rule based on four CA parameters (S , AA , CE , Z_{right}) must be evaluated in larger rule spaces. The problem in tackle is not the discard

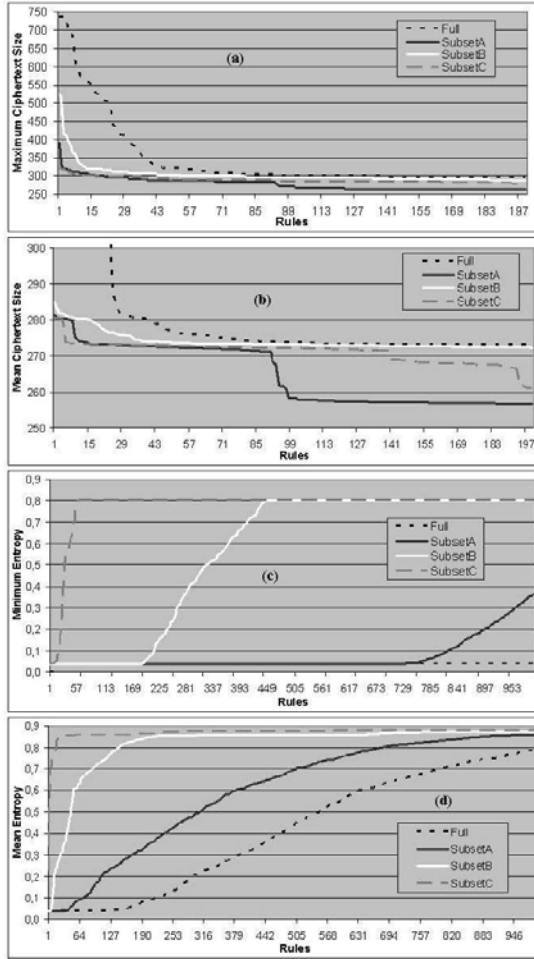


Fig. 1. Underperforming rules for each set: a) Maximum ciphertext size: 200 worst rules. b) Mean ciphertext size: 200 worst rules. c) Minimum D' entropy: 1000 worst rules. d) Mean D' entropy: 1000 worst rules.

of possible keys reducing the available key space. The challenge is how to characterize this behavior aiming to recognize any avoidable secret key in an arbitrary radius CA rule space. The next space is the radius 3 toggle rules formed by approximate 265 potential keys and an exhaustive analysis of them is clear unviable. A final test was performed using radius 3 rules with $Z_{right} = 1$ aiming to evaluate if the filter rule can also be applied to radius 3 rule space. A new sample with 3,000 right-toggle radius 3 rules attending the filter rule was used. We employed them using the VLE environment to cipher a hundred 256-bits plaintexts, by calculating 32 consecutive pre-images. Considering all radius 3 rules, we obtained the following average values: for 256-bits plaintexts, $L_{mean} = 267.07$,

$F_{mean} = 1.845$, $E_{mean} = 0.87$ and $\sigma_{mean} = 4.33\%$. They are satisfactory mean values for radius 3 rules applied to 256-bits lattices. Moreover, no rule presented a low entropy mean.

6 Final Remarks

We investigated a cryptographic model based on the application of the reverse algorithm [11] in the encryption process and the usage of toggle CA rules as secret keys. The general idea of this method was proposed in a previous work [8]. This method alternates during the ciphering process the employment of the original reverse algorithm [11] with a variation inspired in Gutowitz's model [6], which adds extra bits when a pre-image is calculated. The plaintext is encrypted using this method to calculate P consecutive pre-images using a CA rule τ . As the number of failures when calculating consecutive pre-images using the reverse algorithm is variable, the final ciphertext length can also vary. Starting from the ciphertext, the recipient needs to apply the transition rule τ forward by P steps and the final lattice will be the plaintext. Therefore, this approach is a variable-length encryption method named here as VLE.

The average of standard deviation found is 3.66%, showing this method is very robust to a differential cryptanalysis-like attack being much lower than the upper bound limit suggested in [16]: 10%. Comparing with the results presented in [16] the superiority of VLE is clear in such criteria: 12%, 7% and 5% returned by DES, AES and CAC [16] respectively, being the last one a CA-based method. Besides, the absence of an ordered pattern when ciphering similar plaintexts was evidenced by the mean entropy found: 0.876.

Using VLE we have the guarantee that ciphering is possible even if an unexpected Garden-of-Eden state occurs. However a short length ciphertext depends on the secret key specification. The properly specification of the rules/key was deeper investigated in the present work using all the 65,536 radius 2 right-toggle rules. It became clear that there are some rules in this set inappropriate to be used as secret keys: 1.9% of them returning long plaintexts (in contrast to what is expected for VLE) and 2.7% of them exhibiting a more dangerous behavior, which is not able to encrypt a plaintext. Therefore, some kind of restriction is need. Initially, we investigated specifications based on previous works [8] and [9], which uses static rule parameters Z and S . Our experimental results showed that none of these previous specifications are satisfactory. Thus, in a subsequent phase, we employed an analysis based on several CA static parameters. Therefore, we were able to find a good specification of rules to be used as valid secret keys. This specification had shown to be good to filter the complete set of radius 2 rules and to elaborate new radius 3 rules.

The resultant ciphering method has advantages in relation both to the original reverse algorithm [9] and to Gutowitz's pre-image computation [6]: it has a 100% of guarantee of success when ciphering any initial lattice, while the original reverse algorithm can fail in this process if a Garden-of-Eden state [11] is found; it returns a ciphertext length close or equal to the plaintext, while the pre-image computation in Gutowitz's model returns a significant length increase.

Acknowledgements

GMBO thanks CNPq and FAPEMIG support. The authors also thank UFU/PROPP.

References

1. Wolfram, S.: Cryptography with cellular automata. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 429–432. Springer, Heidelberg (1986)
2. Wolfram, S.: Random Sequence Generation by Cellular Automata. *Advances in Applied Mathematics* 7, 123–169 (1986)
3. Tomassini, M., Perrenoud, M.: Stream Ciphers with One and Two-Dimensional Cellular Automata. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 722–731. Springer, Heidelberg (2000)
4. Benkiniouar, M., Benmohamed, M.: Cellular Automata for Cryptosystem. In: Proceedings of IEEE Conference Information and Communication Technologies: From Theory to Applications, pp. 423–424 (2004)
5. Nandi, S., Kar, B., Chaudhuri, P.: Theory and Applications of CA Automata in Cryptography. *IEEE Transactions on Computers* 43, 1346–1357 (1994)
6. Gutowitz, H.: Cryptography with Dynamical Systems. In: Goles, E., Boccara, N. (eds.) *Cellular Automata and Cooperative Phenomena*, vol. 1, pp. 237–274. Kluwer Academic Press, Dordrecht (1995)
7. Oliveira, G., Coelho, A., Monteiro, L.: Cellular Automata Cryptographic Model Based on Bi-Directional Toggle Rules. I. *J. Modern Physics C* 15, 1061–1068 (2004)
8. Oliveira, G., Macêdo, H., Branquinho, A., Lima, M.: A cryptographic model based on the pre-image computation of cellular automata. In: *Automata-2008: Theory and Applications of Cellular Automata*, pp. 139–155. Luniver Press (2008)
9. Wuensche, A.: Encryption using cellular automata chain-rules. In: Adamatzky, et al. (eds.) *Automata-2008: Theory and Applications of Cellular Automata*, pp. 126–138. Luniver (2008)
10. Oliveira, G.M.B., Martins, L.G.A., Alt, L.S., Ferreira, G.B.: Investigating a Cellular Automata-Based Cryptographic Model with a Variable-Length Ciphertext. In: *CSC 2010 - International Conference on Scientific Computing*, Las Vegas (2010)
11. Wuensche, A., Lesser, M.: *Global Dynamics of Cellular Automata*. Addison-Wesley, New Mexico (1992) ISBN: 0-201-55740-1
12. Langton, C.: Computation at the Edge of Chaos: Phase Transitions and Emergent Computation. *Physica D* 42, 12–37 (1990)
13. Oliveira, G., de Oliveira, P., Omar, N.: Definition and applications of a five-parameter characterization of 1D cellular automata rule space. *Artificial Life* 7(3), 277–301 (2001)
14. Wuensche, A.: Classifying Cellular Automata Automatically: Finding gliders, filtering, and relating space-time patterns, attractor basins and the Z parameter. *Complexity* 4(3), 47–66 (1999)
15. Sen, S., Shaw, C., Chowdhuri, D., Ganguly, N., Chaudhuri, P.: Cellular Automata based Cryptosystem (CAC). In: Deng, R.H., Qing, S., Bao, F., Zhou, J. (eds.) *ICICS 2002*. LNCS, vol. 2513, pp. 303–314. Springer, Heidelberg (2002)
16. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) *CRYPTO 1990*. LNCS, vol. 537, pp. 2–21. Springer, Heidelberg (1991)
17. Stallings, W.: *Cryptography and Network Security: Principles and Practice*. Prentice Hall, Englewood Cliffs (2003), ISBN:0-13-091429-0

Network Decontamination with Temporal Immunity by Cellular Automata*

Yassine Daadaa, Paola Flocchini, and Nejib Zaguia

SITE, University of Ottawa, Ottawa, ON K1N 6N5, Canada
{ydaadaa,flocchin,zaguia}@site.uottawa.ca

Abstract. Network decontamination (or disinfection) is a widely studied problem in distributed computing. Network sites are assumed to be contaminated (e.g., by a virus) and a team of agents is deployed to decontaminate the whole network. In the vast literature a variety of assumptions are made on the power of the agents, which can typically communicate, exchange information, remember the past, etc.

In this paper we consider the problem in a much weaker setting; in fact we wish to describe the global disinfection process by a set of cellular automata local rules without the use of active agents. We consider the grid, which is naturally described by a 2-dimensional cellular automata, and we devise disinfection rules both in the common situation where after being disinfected a cell is prone to re-contamination by contact, and in a new setting where disinfection leaves the cells immune to recontamination for a certain amount of time (*temporal immunity*). We also distinguish between Von Neuman and Moore neighborhood, showing that, not surprisingly, a bigger neighborhood allows for a more efficient disinfection.

1 Introduction

1.1 The Problem

Consider a network where nodes are processing performing some computations, but where some of them might be contaminated (e.g., by a virus). Once contaminated, a node might behave incorrectly; furthermore it could cause its neighboring cells to become contaminated as well, thus propagating faulty computations. A pressing concern for fault tolerance and security is obviously to devise strategies to correct the faulty behavior at network sites and to neutralize the propagation of faults. To this end, nodes are endowed with antiviral software that can be activated to perform local disinfection leaving the node clean and possibly immune to further contamination for a certain amount of time. When a node is disinfected however, there is no guarantee it will not be contaminated again; when the immunity time expires in fact, it stays unprotected and becomes contaminated if any neighbour is. The problem is that nodes cannot detect the presence of a virus in themselves and in their neighborhood, and they cannot

* This work was partially supported by NSERC.

control the spread of contamination. They can however detect whether the antiviral process is active or has been activated in their neighborhood; moreover, they want to sparingly activate the anti-virus because, when active, it interrupts any other local computation (e.g., the trivial solution of simultaneously running the anti-virus in all nodes would be unfeasible as it would completely disrupt the computation). The goal is to devise a strategy that makes all nodes simultaneously clean, regardless of the initial contamination pattern, minimizing the number of nodes simultaneously running the anti-virus at any time.

Interestingly, this process can be modelled using a cellular automaton where different rules correspond to different strategies. At any point in time a node can be in one of the following three states: $\{\textit{disinfecting}, \textit{disinfected}, \textit{unprotected}\}$. A *disinfecting* node is running the antiviral software that will leave the node *disinfected* (clean and protected for some time t which depends on the the antiviral properties); a *disinfected* node has run the anti-viral software; an *unprotected* node has never run the antiviral software. Both *disinfected* and *unprotected* nodes are processing and running the underlying computation. In particular, we will consider two cases regarding the local disinfecting procedure, which could leave the node: 1) disinfected but with no immunity (*basic disinfection*), 2) disinfected and immunized for a certain amount of time (*temporal disinfection*). In this context, the goal is to start the disinfection process by setting the state of some cells to *disinfecting* and letting *unprotected* cells activate disinfection on the basis of local rules so that, even in the worst case when all cells are initially contaminated, at the end of the process they are all simultaneously clean. Note that, when all nodes are simultaneously clean they will stay clean forever as we assume contamination is already present before the disinfection begins, and cannot “enter” the system. We are interested in minimizing the number of simultaneous disinfecting sites for a given immunity time or, conversely, minimizing the immunity time for a given number of simultaneous disinfecting sites. Moreover, we are interested in monotone disinfection strategies, where the anti-virus is run only once at each node, that is, once a node is *disinfected*, we must guarantee that, regardless of its protection level, it will stay clean forever.

We consider a common network topology: the grid, which naturally corresponds to a 2-dimensional cellular automaton $n \times n$. We first look at *basic* disinfection and we describe a simple strategy that employs the minimum possible number of disinfecting sites $k = n$ per time unit. We then turn to *temporal* disinfection and we show that, with Von Neumann neighborhood, disinfection can be achieved with $k = 1$ disinfecting site per time unit *if and only if* the immunity time is at least $4(n - 1) - 1$. We also extend the technique for $k = 2$ and $k = 4$. We then consider Moore neighborhoods. Also in this case we describe optimal disinfection rules for the case $k = 1$ and immunity time at least $2n - 1$. The same set of rules allows disinfection with $k > 1$ simultaneously disinfecting sites and immunity time at least $\lfloor \frac{2(2n-1)}{k} \rfloor$ (see Table II).

Due to lack of space some proofs are sketched and some omitted.

Table 1. Summary of results

Neighborhood	Number of disinfecting sites	Immunity time
Von Neumann	$k = 1, 2, 4$	$\geq \frac{4}{k}(n - 1) - 1$
Moore	k	$\geq \lfloor \frac{2(2n-1)}{k} \rfloor$

1.2 Related Work

A large body of work exists on the network decontamination problem using a team of mobile agents, which appropriately move from node to neighbouring node to activate the disinfection (e.g., see [12,4,8,11]). In such contexts, the goal is typically to devise a strategy for the agents to collaboratively decontaminate the whole network using the smallest possible team in such a way that, once cleaned, a node does not get recontaminated. In the existing work, agents are regulated by a variety of assumptions. For example, agents usually can communicate with each other, sometimes in a face-to-face manner (i.e., when they meet at a node) or by writing messages for each other on special spaces located at the nodes; agents have local memory where they store information about their past actions or about a map of the network; agents are usually identified by distinct ids, they are sometimes coordinated by a special agent that acts as a leader, and they might be able to clone themselves. With the exception of [6,11], no immunity has ever been considered. In [11] immunity is defined differently and is related to the number of contaminated neighbours; in [6] it is defined as in this paper but it is assumed in a different setting and exclusively for the case of trees. Determining the minimum size of a decontamination team, even without any immunity, is known to be NP-hard ([12]); optimal strategies have been devised for special topologies (typical topologies are the grid, the torus, the hypercube, the chordal ring), and studies on the various models have been done to describe the computational relationship between them (e.g., see [12,4,8,10]). In [3] decontamination by mobile agents has been linked to cellular automata, but in a very different way: contamination was regulated by cellular automata rules (either unanimity or majority rules), decontamination (called in that context *external decontamination*) was instead performed by mobile agents.

2 Definitions

A 2-dimensional cellular automata (CA) can be described by a quadruple $C = \langle \mathbb{Z}^2, \{0, 1, \bullet\}, N, f \rangle$ where: \mathbb{Z}^2 represents the set of *cells* (also called *sites* or *nodes*); $\{0, 1, \bullet\}$ is the set of *states* of the cells; N is the *neighbourhood* of a cell, and $f : \{0, 1, \bullet\}^{|N|} \rightarrow \{0, 1, \bullet\}$ is the *local transition rule* (or simply *local rule*) of the automaton. In the following we will be considering both *von Neumann* and *Moore* neighborhoods at distance one. Given a cell (i, j) , the *von Neumann* neighborhood consists of the cell itself, plus the four cells at distance one in the Manhattan norm. The *Moore* neighborhood, besides considering the cells of the von Neumann neighborhood, also includes the four neighboring “diagonal cells”.

Given an initial configuration, C^0 , that is a mapping $C^0 : \mathbb{Z}^2 \rightarrow \{0, 1, \bullet\}$, cell states are synchronously updated at each time step by the local transition rule applied to their neighbourhoods. A configuration is the resulting map $C^t : \mathbb{Z}^2 \rightarrow \{0, 1, \bullet\}$ at any time t . In the following we will denote a transition rule by indicating the neighboring states in clock-wise order starting from the left neighbour, for example, in the case of Von Neumann neighborhoods we will use the following notation: $(x_{i,j}^t, x_{i-1,j}^t, x_{i,j+1}^t, x_{i+1,j}^t, x_{i,j-1}^t) \rightarrow x_{i,j}^{t+1}$, and we will denote an arbitrary state by an asterisk (*). A *finite* 2-dimensional Boolean cellular automaton has a finite number of non-zero states in an infinite quiescent background. That is, $C^t(z) = 0$ for all but finitely many $z \in \mathbb{Z}^d$. In this case, let $n \times n$ be the size of the finite lattice initially containing non-zero states; moreover, let us indicate as (i, j) a cell in column i , row j with respect to the finite lattice indicating the left-bottom corner with $(0, 0)$. *Border* cells are cells $(i, n-1), (0, j), (n-1, j)$ with $0 < i, j < n-1$, the four *corner* cells are $(0, 0), (0, n-1), (n-1, 0), (n-1, n-1)$, all other cells are called *internal*. A *Circular* cellular automata can be thought of a 2-dimensional grid where the last node of a row (resp. column) is connected to the first.

3 Basic Disinfection

In this section we consider basic disinfection in finite 2-dimensional CAs, where the disinfecting process does not provide any type of immunity and a cell could be contaminated as soon as one of its neighbours is. In this case any disinfection has to forbid a *disinfected* cell to ever be in contact with a *unprotected* one, which is potentially contaminated.

Notice that with basic disinfection, a single disinfecting site per time unit would obviously not be sufficient to perform decontamination because immediately after becoming *disinfected* a site would inevitably be exposed to a *unprotected* site as at most one of its neighbors could become *disinfecting*. The question is what is the minimum number of disinfecting sites which could guarantee disinfection without recontamination. We prove in the following that n disinfecting sites are necessary and sufficient.

Theorem 1. *Optimal basic disinfection can be achieved in a finite CA with Von Neumann neighborhood using n simultaneous disinfecting sites.*

Proof (Sketch). The proof that n is sufficient is constructive. The idea is to place the n initial disinfecting sites in the cells of the first column of the CA and to have the cells act according to the following very simple rules (see Table 2): regardless of the neighbouring cells' state, a *disinfecting* cell becomes *disinfected*, an *unprotected* cell becomes *disinfecting* at time $t+1$ if its left neighbour is *disinfecting* at time t . The effect of the local rules is the sequential disinfection of columns. In fact, it is easy to see by induction on the time steps that the CA is fully disinfected in n time steps with no *disinfected* site ever in contact with an *unprotected* one.

Table 2. Basic disinfection in Finite CAs with Von Neumann neighborhood. All missing combinations of states do leave the cell unchanged.

BASIC-FINITE
Initial disinfecting sites: $(0, j), 0 \leq j \leq n - 1$

Configuration	Next State
$\{\bullet, *, *, *, *\}$	0
$\{1, \bullet, *, *, *\}$	\bullet

To prove optimality, consider a time t during the disinfection process when there are h disinfected cells. To avoid recontamination at time t , these cells must be surrounded by disinfecting cells (or by quiescent cells outside the border of the CA); we will call such cells *protective cells*. It is easy to see that, if the disinfected cells form separate continuous blocks, the amount of necessary protective cells is never smaller than if the disinfected cells belonged to a single block. Moreover, it has been shown in [9] that for a block of size h at least $\min\{n, \lfloor \frac{1+\sqrt{1+8h}}{2} \rfloor\}$ protective cells are necessary. Since $\lfloor \frac{1+\sqrt{1+8h}}{2} \rfloor < n$ implies $h < \lfloor \frac{n(n-1)}{2} \rfloor$, we have that less than n simultaneous disinfecting cells can protect less than half the cells of the CA, which then cannot be fully disinfected.

4 Temporal Disinfection

The case of temporal disinfection is quite different and more interesting. We must design a set of local rules and choose the location of the initial *disinfecting* sites in such a way that during the evolution of the CA, a *disinfected* node never comes into contact with an *unprotected* one after its immunity time has expired. We distinguish here the two types of neighborhood: Von Neumann and Moore, noticing that the amount of influence from neighbouring cells highly impacts the efficiency solutions.

4.1 Von Neumann Neighbourhood

We first show that, if we impose the use of a single disinfecting cell per time unit, temporal disinfection can be achieved if and only if the immunity time is $\geq 4(n - 1) - 1$. In the following, and in the rest of the paper, when we say that disinfection “propagates”, we indicate that a *disinfecting* cell becomes *disinfected* and one or more of its neighbours become *disinfecting*, thus simulating the propagation of disinfection from cell to neighbouring cell(s).

Theorem 2. *With a single disinfecting site per time unit, temporal disinfection is possible if and only if the immunity time is $\geq 4(n - 1) - 1$.*

Proof (Sketch). Disinfection is achieved by placing the initial disinfection site at a corner and by applying the set of rules indicated in Table 3 (a) (SINGLE-TEMPORAL-VN). It is easy to see that the effect of the local rules is the spiral

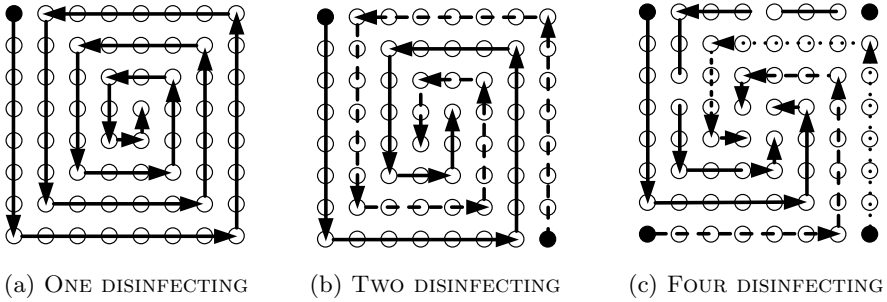


Fig. 1. Propagation of disinfection with temporal immunity and Von Neumann neighborhood

propagation of the disinfecting site (see Figure 1) where a *disinfected* cell is never in contact with an *unprotected* one for more than $4(n-1)-1$ time units, resulting in a situation where all nodes are simultaneously clean after $n \times n$ time units.

We now show that there exist no initial placement for which a correct set of rules exists when the immunity time smaller than $4(n-1)-1$, where a correct set of rules is such that it maintains a single disinfecting site per time unit, and eventually disinfects the whole network never leaving a disinfected site in contact with an *unprotected* one for more than $4(n-1)-1$ time units after it has been cleaned.

We have three possible placements of the initial disinfecting site: corner, border, and internal. Let us first consider the case when the initial disinfecting site is internal, in this case we will show that, regardless of the immunity time, there exist no set of rules that allows to maintain a single disinfecting site. Let us call *forbidden rule* a rule that cannot be present in a correct set of rules. First of all notice that $r_0 = \{\bullet, *, *, *, *\} \rightarrow 0$ is necessarily part of the set of rules to insure the presence of a single disinfecting site. Moreover, to allow the spread of disinfection to a single new site, one (and only one) of the following four rules must be present ($r_1 = \{1, \bullet, 1, 1, 1\} \rightarrow \bullet$, $r_2 = \{1, 1, \bullet, 1, 1\} \rightarrow \bullet$, $r_3 = \{1, 1, 1, \bullet, 1\} \rightarrow \bullet$, $r_4 = \{1, 1, 1, 1, \bullet\} \rightarrow \bullet$). Without loss of generality, let r_1 be present and let r_2, r_3, r_4 be forbidden. The combination of r_0 and r_1 makes the disinfecting site propagate to the right until reaching the border. At this point, for insuring the propagation of the disinfecting site, one (and only one) of the following three rules must be added to the set: $r_5 = \{0, 0, 1, \bullet, 1\} \rightarrow \bullet$, $r_6 = \{1, 1, 1, 0, \bullet\} \rightarrow \bullet$, $r_7 = \{1, 1, \bullet, 0, 1\} \rightarrow \bullet$. We can show that r_5 is forbidden (the disinfecting site would propagate back to the original site where however, because of the fact that $r_2, r_3,$ and r_4 are forbidden the only possible movement would be to the right again giving rise to an oscillation). Let, w.l.g, r_6 be present and r_7 be forbidden. Because of r_6 being present, rule $r_8 = \{1, 1, 1, \bullet, 0\} \rightarrow \bullet$ must be forbidden, otherwise more than one site would be in disinfecting state. The combination of r_0 and r_6 makes the disinfecting site propagating up until reaching the down neighbour of the corner, at this point, since r_3 is forbidden, we need to add a rule to ensure propagation, and the only possible one is $r_9 = \{1, 1, 0, 0, \bullet\} \rightarrow \bullet$ and

Table 3. Rule Tables for 1/2/3 disinfecting sites in a Finite CA with temporal immunity and Von Neumann neighborhood. All missing combinations of states for the neighbourhood of cell (i, j) do leave $x_{i,j}$ unchanged.

SINGLE-TEMPORAL-VN		TWO-TEMPORAL-VN		FOUR-TEMPORAL-VN	
Init: a corner		Init: two opposite corners		Init: all corners	
Configuration	Next State	Configuration	Next State	Configuration	Next State
{•, *, *, *, *}	0	{•, *, *, *, *}	0	{•, *, *, *, *}	0
{1, •, 1, 1, 0}	•	{1, •, 1, 1, 0}	•	{1, •, 1, 1, 0}	•
{1, •, 1, 0, 0}	•	{1, •, 1, 0, 0}	•	{1, 0, •, 1, 1}	•
{1, 0, •, 1, 1}	•	{1, 0, •, 1, 1}	•	{1, 0, •, 1, 0}	•
{1, 0, •, 1, 0}	•	{1, 0, •, 1, 0}	•	{1, 0, •, 0, 0}	•
{1, 0, •, 0, 0}	•	{1, 0, •, 0, 0}	•	{1, 0, 0, •, 1}	•
{1, 0, 0, •, 1}	•	{1, 1, 0, •, 1}	•	{1, 1, 0, •, 1}	•
{1, 1, 0, •, 1}	•	{1, 0, 0, •, 1}	•	{1, 1, 1, 0, •}	•
{1, 1, 1, 0, •}	•	{1, 1, 1, 0, •}	•	{1, 1, 0, 0, •}	•
{1, 1, 0, 0, •}	•	{1, 1, 0, 0, •}	•	{1, •, 1, •, 0}	•
{1, 1, 0, 0, •}	•	{1, •, 0, 0, •}	•	{1, •, 0, •, 1}	•
{1, 0, 0, 0, •}	•	{1, 0, •, •, 0}	•	{1, 0, •, 1, •}	•
		{1, 0, •, 0, •}	•	{1, 1, •, 0, •}	•
		{1, 0, •, 0, •}	•	{1, •, •, •, •}	•

the disinfection will reach the corner. At this point, for insuring the propagation of the disinfecting site, one (and only one), of the following two rules must be added to the set: $(r_{10} = \{0, 1, \bullet, 0, 0\} \rightarrow \bullet)$, and $r_{11} = \{1, 1, 0, \bullet, 1\} \rightarrow \bullet$. Rule r_{10} is forbidden because the disinfecting site would propagate back to the the first disinfected site on the border and only an oscillation will be permitted (all the other rules are forbidden). Following a similar reasoning, one can see that the disinfecting site is forced to propagate left until reaching the corner with the necessary inclusion of rule $r_{12} = \{1, 0, 0, \bullet, 1\} \rightarrow \bullet$, then down following another necessary rule $r_{14} = \{1, 0, \bullet, 1, 1\} \rightarrow \bullet$. At the next step, however two sites $((0, n - 3)$ and $(1, n - 2))$ will become simultaneously disinfecting (due to rules r_1 and r_{14} respectively), which contradicts our hypothesis. It follows that starting from an internal cells we cannot decontaminate all sites with a single disinfecting site per time unit.

If the initial disinfecting site is a border site or a corner, one can show, using similar arguments, that all combinations of possible rules either lead to an impossibility, or to a disinfection requiring an immunity higher than $4(n - 1) - 1$.

Interestingly, the strategy can be generalized when we allow two or four simultaneous disinfecting sites. In the first case we achieve disinfection initially placing the two disinfecting sites in two opposite corners and having the rules of Table 3 (b) describe the local behavior of the cells; in the second case the four disinfecting

sites are initially placed in the four corners and the rules are described in Table 3 (c). The global behavior of the automata corresponds to the propagation of the disinfection in spirals (see Figure 1 b) and c)), and it is easy to see that the immunity time is at least $2(n - 1) - 1$ for the case of two disinfecting sites, and $(n - 1) - 1$ for four. In fact, we can then conclude:

Theorem 3. *With $k = 1, 2, 4$ disinfecting site per time unit, temporal disinfection is possible if and only if the immunity time is $\geq (4 - k)(n - 1) - 1$.*

4.2 Moore Neighbourhood

The bounds are different if we increase the neighborhood to include diagonal neighbours. In fact, we can show that with a single disinfecting site optimality is reached when the immunity time is at least $2n - 1$. We now describe a general set of rules which achieves optimality in the particular case of a single agent placed on a corner, but which works correctly also with more disinfecting sites.

Table 4. Rule Table for Finite CA with temporal immunity and Moore neighborhood. All missing combinations of states for the neighbourhood of cell (i, j) do leave $x_{i,j}$ unchanged.

TEMPORAL-MOORE			
Configuration	Next State	Configuration	Next State
{•, *, *, *, *, *, *, *, *}	0	{1, •, •, 0, 0, 0, 0, 0, 0}	•
{1, •, 0, 1, 1, 1, 1, 1, 0}	•	{1, 0, 0, •, 0, 0, 0, 0, 0}	•
{1, •, 0, 0, 0, 1, 1, 0, 0}	•	{1, 0, 0, •, 1, 1, 1, 1, 0}	•
{1, •, 0, 1, 1, 1, 0, 0, 0}	•	{1, 0, 0, •, 1, 1, 0, 0, 0}	•
{1, •, 0, 0, 0, 1, 1, 1, 0}	•	{1, 0, 0, •, 0, 0, 0, 1, 0}	•
{1, •, 0, 1, 1, 1, 1, 1, •}	•	{1, 0, 0, •, 1, 1, 1, •, 0}	•
{1, •, 0, 1, 0, 0, 0, 1, 0}	•	{1, 0, 0, •, 0, 0, 0, •, 0}	•
{1, •, 0, 1, 0, 0, 0, 0, 0}	•	{1, 0, 0, 1, 0, 0, 0, •, 0}	•
{1, •, 0, 0, 0, 0, 0, 1, 0}	•	{1, 0, 0, 0, 0, 0, 0, •, 0}	•
{1, •, •, 1, 1, 1, 1, 1, 0}	•	{1, 0, 0, 0, 0, 1, 1, •, 0}	•
{1, •, 0, 1, 0, 0, 0, 1, •}	•	{1, 0, 0, 1, 1, 1, 1, •, 0}	•
{1, •, •, 1, 0, 0, 0, 1, 0}	•		

Any number of *disinfecting* sites (greater than 1) are initially placed on the first column at distance greater than 1 from each other (i.e, no two consecutive cells are initially disinfecting). Careful inspection of the rules in Table 4 shows that disinfection “propagates” vertically to unprotected cells (an *unprotected* cell becomes *disinfecting* when its down/up neighbour, or both, are *disinfecting*) until they reach either another disinfecting cell or a clean one, in which case they “propagate” to the next column (an *unprotected* cell becomes *disinfecting* when its left neighbour is *disinfecting* and at least one of its left diagonal left neighbours are *disinfected*). Figure 2 shows the effect of the local rules on some partial configurations. Figure 3 describes instead the global propagation path of the disinfection. In the following we prove the correctness of this set of rules.

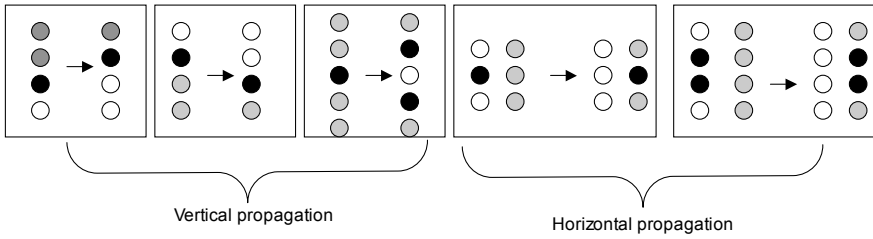


Fig. 2. Examples of vertical and horizontal propagation. Black cells are *disinfecting*, grey cells are *unprotected*, white cells are *disinfected*.

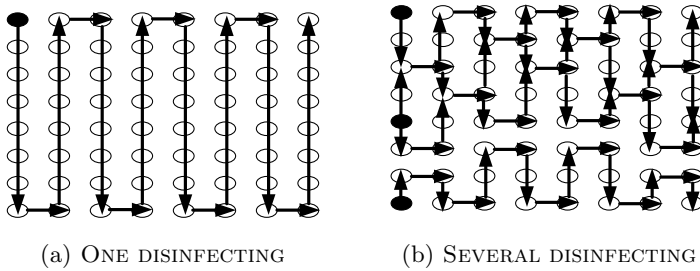


Fig. 3. Propagation of disinfection with temporal immunity and Moore neighborhood

Theorem 4. Let d_{max} be the maximum distance between two consecutive disinfecting cells or between a disinfecting cell and a corner at time 0. In the initial number of disinfecting sites is at least 2, the set of rules TEMPORAL-MOORE achieves disinfection with immunity time $t = d_{max}$.

Proof (Sketch). To prove the theorem, we need to prove that disinfection is achieved monotonically. That is, that once *disinfected*, every cell stays clean until the end of the process, when all cells are clean. Let us call *entry points* of disinfection in a column, the cells in such a column that become disinfecting due to a left disinfecting neighbour (horizontal propagation). We now prove by induction on the number of columns that, for each column i there is a time t when: (i) all cells in column i are either *disinfected* or *disinfecting*; (ii) all cells in column $i - 1$ (for $i > 0$) are clean; (iii) by time $t + d_{max}$ all right neighbours of a *disinfected* cell of column i are either *disinfected* or *disinfecting*; (iv) the distance between any two consecutive entry points in column $i + 1$ (for $i < n - 1$) is smaller than d_{max} .

1. Base - column 0: According to the set of rules TEMPORAL-MOORE the *disinfecting* state propagates vertically in both directions on the first column, and since the maximal distance between initially consecutive disinfecting cells is d_{max} by construction, within $\lfloor \frac{d_{max}-1}{2} \rfloor$ time units all the cells on column 0 are either *disinfecting* or *disinfected*. According to the local rules, disinfection then propagates to column 1. Since the propagation to column 1 happened for all

disinfecting cells within $\lfloor \frac{d_{max}-1}{2} \rfloor$ time units from the beginning, the distance between any two consecutive entry points in column 1 cannot be greater than d_{max} . By a similar argument as for column 0, all cells in column 1 will then become *disinfected* or *disinfecting* within other $\lfloor \frac{d_{max}-1}{2} \rfloor$ time units. Thus, within at most d_{max} time units, all the cells in column 0 and in column 1 will be either *disinfected* or *disinfecting*.

2. Induction hypothesis: At some point during the computation, assume all cells of column i ($0 < i < n - 1$) and all their right neighbours in column $i + 1$ are either in *disinfecting* or *disinfected* state, their left neighbours in column $i - 1$ are clean, and the entry points in column $i + 1$ are at distance at most d_{max} .

3. Induction Step: Consider column $i + 1$. By induction hypothesis we know that there is a time t when all cells in column $i - 1$, are clean, the ones in columns i , and $i + 1$ are either *disinfecting* or *disinfected* and the entry points in column $i + 1$ are at maximum distance d_{max} from each other. It follows that the disinfection propagates vertically in column $i + 1$ within $\lfloor \frac{d_{max}-1}{2} \rfloor$ time units from time t , thus leaving all cells in column i clean. Disinfection propagates then to column $i + 2$ and within other $\lfloor \frac{d_{max}-1}{2} \rfloor$ time units all the cells in the column $i + 2$ become either disinfected or disinfecting. We can conclude that, by time $t + d_{max}$ all cells in column $i + 1$ together with all their right neighbours are either *disinfected* or *disinfecting* and the cells in column i are clean, thus concluding the proof.

Placing $k > 1$ disinfecting sites roughly equidistant on the first column we obtain as a corollary that:

Corollary 1. *With $k > 1$ disinfecting site per time unit, temporal disinfection can be achieved when the immunity time is at least $\lfloor \frac{2(2n-1)}{k} \rfloor$.*

For the case of $k = 1$ when the starting disinfecting cell is a corner, with an argument similar to the one of Theorem 2 we can show that the strategy is optimal:

Theorem 5. *With a single disinfecting site per time unit, temporal disinfection is possible if and only if the immunity time is at least $2n - 1$.*

5 Note on Circular CAs

In this Section we briefly discuss how the results for finite CAs can be extended to circular CAs.

In the case of *basic disinfection*, the decontamination technique can be easily extended to circular CAs with Von Neumann neighborhood with the same immunity time, doubling the number of simultaneously disinfecting sites. The n initial disinfecting sites are placed in the cells of any column and the cells obey the following simple rules: regardless of the neighbouring sites' state, a *disinfecting* site becomes disinfected; an *unprotected* site becomes *disinfecting* at time $t + 1$ if one or both its horizontal neighbours (left/right) are *disinfecting* at time t . It is easy to see that the effect of the local rules is the sequential

disinfection of pairs of columns, where a disinfected site is never in contact with an unprotected one, and we have:

Theorem 6. *Basic disinfection can be achieved in a circular CA with Von Neumann neighborhood using $2n$ simultaneous disinfecting sites.*

Also in the case of *temporal disinfection* in CAs with Moore neighbourhood we can easily modify the strategy employed for the finite case to obtain disinfection with the same immunity time, doubling the number of simultaneously disinfecting sites. We place the initial disinfecting cells on the first column (at distance greater than 1 from each other). We design the rules in such a way that the disinfection propagates in *both* directions (up and down) on the first column until two disinfecting nodes become adjacent (or a disinfecting node has the two neighbours on the column disinfected). At this point the rules will let the disinfection propagate in both directions (right and left) to the adjacent columns. The procedure continues like for the finite case, but in both directions until two disinfected columns become adjacent or a column has both left and right column neighbours disinfected. The rules are described in Table 5.

Table 5. Rule Table for Circular CA with temporal immunity and Moore neighborhood. All missing combinations of states for the neighbourhood of cell (i, j) do leave $x_{i,j}$ unchanged.

TEMPORAL-CIRCULAR-MOORE		Configuration		Next State	
Configuration	Next State	Configuration	Next State	Configuration	Next State
{•, *, *, *, *, *, *, *, *}	0	{1, 0, 0, 1, •, •, 0, 1, •}	•		
{1, 1, 1, •, 1, 1, 1, 1, 1}	•	{1, 0, 0, 1, 0, •, 0, 1, •}	•		
{1, 1, 1, 1, 1, 1, 1, •, 1}	•	{1, 0, 0, •, 1, •, 0, 1, 0}	•		
{1, 1, 1, •, 1, 1, 1, •, 1}	•	{1, 0, 0, 1, 0, •, 1, •, 0}	•		
{1, •, 0, 1, 1, 1, 1, 1, •}	•	{1, •, 1, •, 0, 0, 0, 1, 0}	•		
{1, •, 0, 1, 1, 1, 1, 1, 0}	•	{1, 0, 0, 1, •, •, 0, 1, •}	•		
{1, 1, 1, 1, 0, •, 0, 1, 1}	•	{1, •, 0, 1, 0, 0, 0, 1, 1}	•		
{1, 1, 1, 1, •, •, 0, 1, 1}	•	{1, 0, 0, 1, •, 1, 1, •, 0}	•		
{1, 0, 0, •, 1, 1, 1, 1, 0}	•	{1, 1, •, 1, 0, 0, 0, •, 1}	•		
{1, 0, 0, 1, 1, 1, 1, •, 0}	•	{1, •, 0, 1, 0, •, 0, 1, 0}	•		
{1, 0, 0, •, 1, 1, 1, •, 0}	•	{1, 0, 0, •, 0, 0, 0, 1, 0}	•		
{1, 1, 1, 1, 0, 0, 0, •, 1}	•	{1, 0, 0, 1, 0, 0, 0, •, 0}	•		
{1, 1, 1, •, 0, 0, 0, 1, 1}	•	{1, 0, 0, •, 0, 0, 0, •, 0}	•		
{1, 1, 1, •, 0, 0, 0, •, 1}	•	{1, 0, 0, 0, 0, 0, 0, •, 0}	•		
{1, •, 0, 1, 0, 0, 0, 1, 0}	•	{1, 0, 0, •, 0, 0, 0, 0, 0}	•		

Let d_{max} be the maximum distance between two consecutive *disinfecting* cells or between a *disinfecting* cell and a corner at time 0. Analogously to the case of finite CAs, it is easy to see that, If the initial number of disinfecting sites is at least 2, the set of rules TEMPORAL-CIRCULAR-MOORE achieves disinfection with immunity time $t = d_{max}$. It then follows that:

Theorem 7. *With $k > 2$ disinfecting sites per time unit, temporal disinfection can be achieved in a circular cellular automata when the immunity time is at least $\lfloor \frac{(2n-1)}{k} \rfloor$.*

Interestingly, for circular CAs with Von Neumann neighborhood, a straightforward extension of the technique described for the finite case cannot be proposed. The symmetry of the CA and the small neighborhood heavily limit the possibilities of disinfection. For example, we can show that:

Theorem 8. *In circular CAs with Von Neumann neighborhood temporal disinfection is not possible with a single disinfecting site per time unit, for any immunity time.*

The general case of $k > 1$ is under investigation.

6 Conclusions

In this paper, we have considered the problem of decontaminating a 2-dimensional 3-states cellular automata with and without temporal immunity. We have focused on finite cellular automata with Von Neumann and Moore neighborhoods. In each case we have described CA rules to achieve decontamination. We have measured the efficiency of our sets of rules by considering the number of simultaneous sites in *disinfecting* state and, with this metric, we have shown that most rules are optimal. We have also observed how to extend the rules to the case of circular CAs.

Some problems are still open and currently under investigation. For example, with Von Neumann neighborhood we have described rules achieving optimal disinfection which employ 1,2, and 4 simultaneously disinfecting sites, but no optimal strategy has been proposed for 3 simultaneously disinfecting sites or for any number between 4 and $n - 1$. In our study we assume that the location of the initial disinfecting sites can be chosen; another interesting problem would be to devise optimal rules for a given arbitrary initial placement. Finally, we are now investigating the case of circular CAs with Von Neumann neighborhood to determine whether, regardless of the amount of temporal immunity, decontamination is possible at all with less than n agents.

References

1. Barrière, L., Flocchini, P., Fraigniaud, P., Santoro, N.: Capture of an intruder by mobile agents. In: Proc. 14th Symp. Parallel Algorithms and Architectures (SPAA 2002), pp. 200–209 (2002)
2. Bienstock, D., Seymour, P.: Monotonicity in graph searching. *J. Algorithms* 12, 239–245 (1991)
3. Flocchini, P.: Contamination and decontamination in majority-based systems. *Journal of Cellular Automata* 4(3), 183–200 (2009)
4. Flocchini, P., Huang, M.J., Luccio, F.L.: Decontamination of hypercubes by mobile agents. *Networks* 52(3), 167–178 (2008)

5. Flocchini, P., Luccio, F.L., Song, L.X.: Size Optimal Strategies for Capturing an Intruder in Mesh Networks. *Communications in Computing*, 200–206 (2005)
6. Flocchini, P., Mans, B., Santoro, N.: Tree decontamination with temporary immunity. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) *ISAAC 2008*. LNCS, vol. 5369, pp. 330–341. Springer, Heidelberg (2008)
7. Fomin, F., Golovach, P.: Graph searching and interval completion. *SIAM J. on Discrete Mathematics* 13(4), 454–464 (2000)
8. Fraigniaud, P., Nisse, N.: Connected treewidth and connected graph searching. In: Correa, J.R., Hevia, A., Kiwi, M. (eds.) *LATIN 2006*. LNCS, vol. 3887, pp. 479–490. Springer, Heidelberg (2006)
9. Qiu, J.: Best Effort Decontamination of Networks. Ms. Thesis, U. of Ottawa (2007)
10. Ilcinkas, D., Nisse, N., Soguet, D.: The cost of monotonicity in distributed graph searching. *Distributed Computing* 22(2), 117–127 (2009)
11. Luccio, F., Pagli, L., Santoro, N.: Network decontamination with local immunization. *International Journal of Foundation of Computer Science* 18(3), 457–474 (2007)
12. Megiddo, N., Hakimi, S., Garey, M., Johnson, D., Papadimitriou, C.: The complexity of searching a graph. *Journal of the ACM* 35(1), 18–44 (1988)

Characterization of CA Rules for SACA Targeting Detection of Faulty Nodes in WSN*

Sukanta Das¹, Nazma N Naskar², Sukanya Mukherjee³,
Mamata Dalui⁴, and Biplab K. Sikdar⁵

¹ Department of Information Technology, Bengal Engineering & Science University,
Shibpur, West Bengal, India, 711103

sukanta@it.becs.ac.in

² Department of Information Technology, Seacom Engineering College, Dhulagarh,
Howrah, India

nazma_preeti@yahoo.co.in

³ Department of Information Technology, Institute of Engineering & Management,
Salt Lake Electronics Complex, Kolkata, India, 700091

⁴ Department of Computer Science & Engineering, National Institute of Technology,
Durgapur, West Bengal, India

mamata.06@gmail.com

⁵ Department of Computer Science & Technology, Bengal Engineering & Science
University, Shibpur, West Bengal, India, 711103

biplab@cs.becs.ac.in

Abstract. The single attractor cellular automata (SACA) is of prime interest in devising schemes for different applications specially in authentication and cryptography. The synthesis of SACA in linear/additive domain has been proposed in literature. This work reports characterization of such a special class of CA beyond linear domain. The characterization is based on the analysis of individual CA rule and its potential to form the single length cycle attractors (point states). The proposed characterization targets design of a CA based scheme for detection of faulty nodes in a wireless sensor network. It enables identification of faults even in multiple nodes with out major computation overhead.

1 Introduction

In 80s, Wolfram [1] studied a family of simple 1-dimensional CA that could simulate complex behaviors [2,3,4,5,6]. The proposed CA structure was viewed as a discrete lattice of 2-state per cell, with 3-neighborhood dependence (self, left and right neighbors). A special class of Wolfram's 3-neighborhood 1-dimensional CA, called the linear/additive CA, had gained immense attention [7].

While characterizing 3-neighborhood CA state space, the researchers identified a set of CA states called attractor towards which neighboring states asymptotically approach in course of dynamic evolution [8]. A single length cycle attractor is one where the number of states of an attractor is one [7].

* This research work is supported by the Sponsored Cellular Automata Research Projects, Bengal Engineering and Science University, Shibpur, WB, India-711103.

The identification of attractors, specially for linear/additive CA, is explored in [7,9,10]. A graph based solution was proposed in [11,12]. However, characterizations of single length cycle attractor for SACA are yet to be explored.

In this context, we concentrate on the characterization of CA rules with the target to construct a single length cycle attractor CA. We then focus on to identify the special class of irreversible CA, referred to as the SACA (single attractor CA), that has great importance in developing the efficient message authentication scheme [13], one way function [14], and other similar applications [7,15]. However, in the present work, we target the design of an SACA based scheme to detect faulty nodes in a wireless sensor network.

2 Cellular Automata Basics

A Cellular Automaton (CA) consists of a number of cells organized in the form of lattice. It evolves in discrete space and time, and can be viewed as an autonomous finite state machine (FSM). Each cell stores a discrete variable at time t that refers to the present state (PS) of the cell. The next state (NS) of the cell at $(t + 1)$ is affected by its state and the states of its neighbors at time t . In this work, we concentrate on such 3-neighborhood CA (self, left and right neighbors), where a CA cell is having two states - 0 or 1 and the next state of i^{th} CA cell is

$$S_i^{t+1} = f_i(S_{i-1}^t, S_i^t, S_{i+1}^t) \tag{1}$$

where S_{i-1}^t , S_i^t and S_{i+1}^t are the present states of the left neighbor, self and right neighbor of the i^{th} cell at time t and f_i is the next state function.

The states of the cells $S^t = (S_1^t, S_2^t, \dots, S_n^t)$ at t is the present state of the CA. Therefore, the next state of an n -cell CA is determined as

$$S^{t+1} = (f_1(S_0^t, S_1^t, S_2^t), f_2(S_1^t, S_2^t, S_3^t), \dots, f_n(S_{n-1}^t, S_n^t, S_{n+1}^t)) \tag{2}$$

The next state function of the i^{th} CA cell can be expressed in the form of a truth table *Table 1*. The decimal equivalent of the 8 outputs is called Rule R_i [23]. In a 2-state 3-neighborhood CA, there can be 2^8 (256) rules. Four such rules 90, 150, 75 and 192 are illustrated in *Table 1*. The first row lists the possible 2^3 (8) combinations of present states of $(i - 1)^{th}$, i^{th} and $(i + 1)^{th}$ cells at t . The last four rows indicate the next states of the i^{th} cell at $(t + 1)$ for different combinations of present states of its neighbors, forming the rules 90, 150, 75 and 192 respectively. The following terminologies are relevant for the current work.

Table 1. RMTs of the CA < 90, 150, 75, 192 >

PS	111	110	101	100	011	010	001	000	Rule
RMT	(7)	(6)	(5)	(4)	(3)	(2)	(1)	(0)	
NS	0	1	0	1	1	0	1	0	90
NS	1	0	0	1	0	1	1	0	150
NS	0	1	0	0	1	0	1	1	75
NS	1	1	0	0	0	0	0	0	192

Definition 1. The set of rules $R = \langle R_1, R_2, \dots, R_i, \dots, R_n \rangle$ that configure the cells of a CA is called the rule vector.

Definition 2. If all the CA cells obey the same rule, then the CA is a uniform CA; otherwise it is a non-uniform/hybrid CA.

Definition 3. A CA is said to be a null boundary CA (NBCA) (Figure 1) if the left (right) neighbor of the leftmost (rightmost) terminal cell is fixed to 0-state.

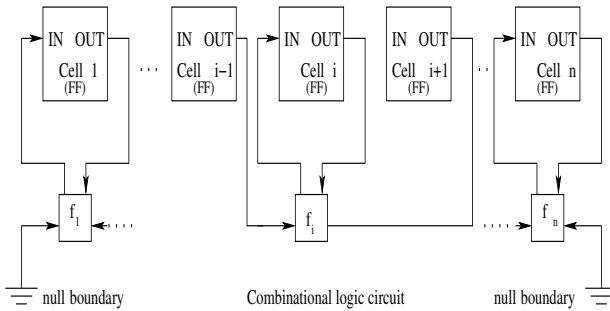


Fig. 1. An n-cell null boundary CA

Definition 4. A CA is reversible if it contains only cyclic states in its state transition diagram (Figure 2); otherwise the CA is irreversible (Figure 3).

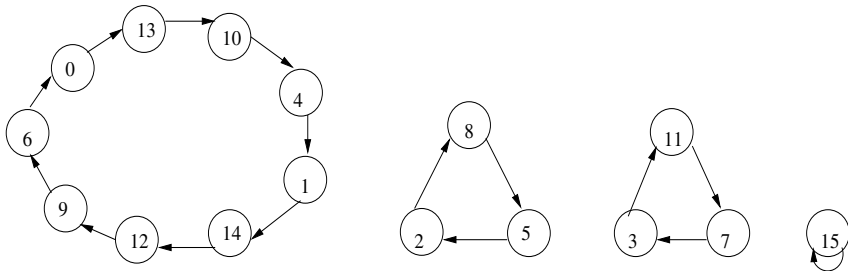


Fig. 2. A 4-cell reversible CA $\langle 105, 177, 170, 75 \rangle$

Definition 5. Rule Mean Term (RMT): From the view point of Switching Theory, a combination of the present states (as noted in the 1st row of Table 1) can be considered as the Min Term of a 3-variable $S_{i-1}^t, S_i^t, S_{i+1}^t$ switching function. Therefore, each column of the first row of Table 1 is referred to as Rule Min Term (RMT). The column 011 of Table 1 is the 3rd RMT. The next states corresponding to this RMT are 1 for Rule 90 and 75, and 0 for Rule 150 and Rule 192.

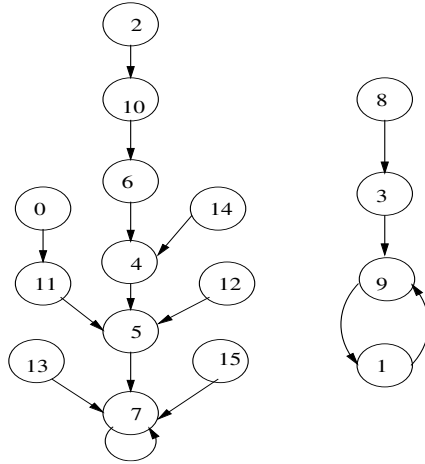


Fig. 3. A 4-cell irreversible CA $\langle 1, 236, 165, 69 \rangle$

Attractor: A set of states forms loop (cycle) in the state transition diagram of a cellular automata ($7 \rightarrow 7$ and $9 \rightarrow 1 \rightarrow 9$ of *Figure 3*) and referred to as the *attractor*. The attractors of single length cycles ($(7 \rightarrow 7$ of *Figure 3*) are of our current interest. In this work we concentrate on *SACA*, where all the states of CA lead to a single attractor of single length cycle.

3 Characterization of CA Rules for SACA

Since the next state of a single cycle attractor is the attractor itself, there should be at least one RMT of each cell rule (R_i) of the CA (R) for which the cell i does not change its state in the next time step. For example, the RMT $x0x$ ($x=0/1$) of a rule is considered to find the next state of cell i when the current states of its left neighbor ($(i - 1)^{th}$ cell), self and right neighbor ($(i + 1)^{th}$ cell) are $x, 0$ and x respectively. It implies, such an RMT is 0, the state change in cell i is $0 \rightarrow 0$. That is, for rule R_i , if the RMT $0(000), 1(001), 4(100), 5(101)$ are 0, then the CA cell i is configured with R_i does not change its state. Similarly, if the value of RMTs $2(010), 3(011), 6(110)$ or $7(111)$ are 1 in a rule R_i , it ensures a cell configured with R_i can stick to its current state in the next time step. When a CA cell is configured with the rule 204, all RMTs of it help the formation of attractors of the CA. On the other hand, RMTs of 51 deny the attractor formation (*Figure 4*).

111	110	101	100	011	010	001	000
1	1	0	0	1	1	0	0

Structure of Rule 204

111	110	101	100	011	010	001	000
0	0	1	1	0	0	1	1

Structure of Rule 51

Fig. 4. Structure of rule 204 and 51

Property 1: A rule \mathcal{R}_i can contribute to the formation of single cycle attractor(s) if at least one of the RMTs 0, 1, 4 or 5 is 0, or the RMTs 2, 3, 6 or 7 is 1.

3.1 Classification of CA Rules

A CA synthesized with arbitrary rules may result in one or more attractors with multi-length cycles (Figure 3). To synthesize an SACA, we classify the 256 CA rules based on Property1. All these 256 rules form 9 groups (group 0-8). The rule 200 (11001000) is in group 7 as it follows Property1 for 7 RMTs. A CA configured with the rules that maintain Property1 for most of the RMTs can increase the chances of single length cycle attractors. Table 2 shows all the nine groups of CA rules. The following observations are the outcome of extensive experimentations.

Observation 1. CA configured with the rules of group 7 or 8 forms only single length cycle MACA.

Example 1. The rule 205 belongs to group 7. The 4-cell uniform CA $< 205, 205, 205, 205 >$ is an MACA and all the attractors are single length cycle (Figure 5). It is observed that the depth of any n -bit uniform CA configured with rule 205 is 2.

Observation 2. Most of the rules of group 6 form single length cycle attractor CA. This is true for 156, 192, 198, and 201. However, some of group 6 rules

Table 2. Classification of CA rule

group	Rule
0	51
1	19, 35, 49, 50, 55, 59, 115, 179
2	3, 17, 18, 23, 27, 33, 34, 39, 43, 48, 53, 54, 57, 58, 63, 83, 99 113, 114, 119, 123, 147, 163, 177, 178, 183, 187, 243
3	1, 2, 7, 11, 16, 21, 22, 25, 26, 31, 32, 37, 38, 41, 42, 47, 52, 56, 61, 62, 67 81, 82, 87, 91, 97, 98, 103, 107, 112, 117, 118, 121, 122, 127, 131 145, 146, 151, 155, 161, 162, 167, 171, 176, 181, 182, 185, 186, 191 211, 227, 241, 242, 247, 251
4	0, 5, 6, 9, 10, 15, 20, 24, 29, 30, 36, 40, 45, 46, 60, 65, 66, 71, 75, 80, 85, 86 89, 90, 95, 96, 101, 102, 105, 106, 111, 116, 120, 125, 126, 129, 130, 135 139, 144, 149, 150, 153, 154, 158, 159, 160, 165, 166, 169, 170, 175 180, 184, 189, 190, 195, 209, 210, 215, 219, 225, 226, 231, 235, 240 245, 246, 249, 250, 255
5	4, 8, 13, 14, 28, 44, 64, 69, 70, 73, 74, 79, 84, 88, 93, 94, 100, 104, 109 110, 124, 128, 133, 134, 137, 138, 143, 148, 152, 157, 164, 168, 173, 174 188, 193, 194, 199, 203, 208, 213, 214, 217, 218, 223, 224, 229, 230, 233 234, 239, 244, 248, 253, 254
6	12, 68, 72, 77, 78, 92, 108, 132, 136, 141, 142, 156, 172, 192, 197, 198 201, 202, 207, 212, 216, 222, 228, 232, 237, 238, 252
7	76, 140, 196, 200, 205, 206, 220, 236
8	204

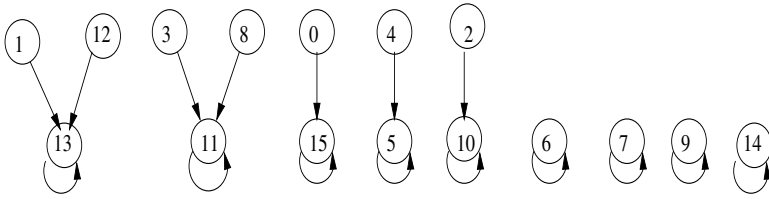


Fig. 5. State transition diagram of $\langle 205, 205, 205, 205 \rangle$

(e.g. 201) form multi graph and some (e.g. 192) form single graph -that is, SACA (Figure 6).

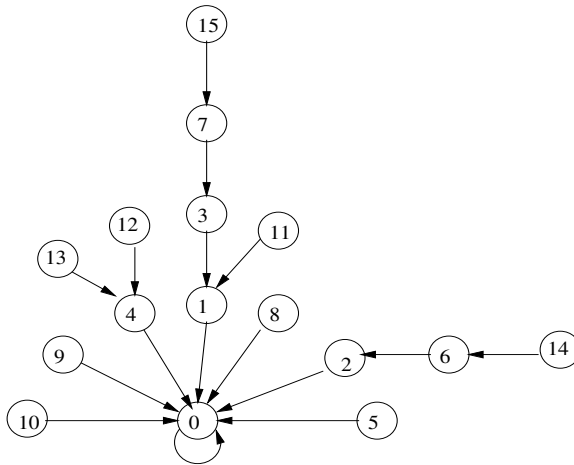


Fig. 6. State transition diagram of $\langle 192, 192, 192, 192 \rangle$

Observation 3. The CA synthesized with rules from group 5 may have multi-length cycle. Such rules are 233, 229, 218, 217, 193, 199, 188, 173, 157, 73 and 28.

Observation 4. Some rules of group 4 form both the single length and multi-length cycles and single and multi-graphs.

Example 2. The state transition diagram of a 4-bit CA is noted in (Figure 7). Here both the single and multi-length cycle attractors are formed and it is multi-graph. The rules that form only single length cycle uniform CA are 0, 10, 15, 20, 24 30, 36, 40, 46, 66, 80, 85, 90, 96, 106, 120, 130, 144, 160, 166, 170, 180, 184, 219, 226, 235, 240, 249, 255. The rules 0, 10, 24, 40, 66, 80, 96, ... form single graph -that is, SACA.

Observation 5. If a CA cell is configured with rule 51 (group 0), the CA forms multi-length cycle attractors only.

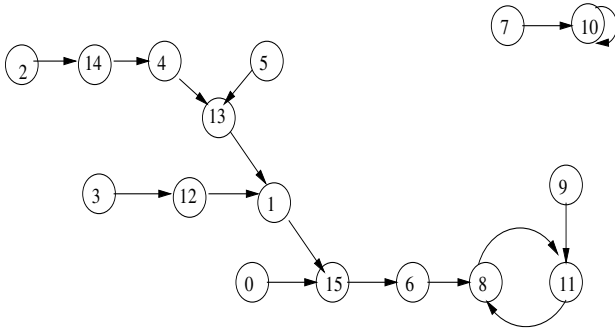


Fig. 7. State transition diagram of $\langle 135, 135, 135, 135 \rangle$

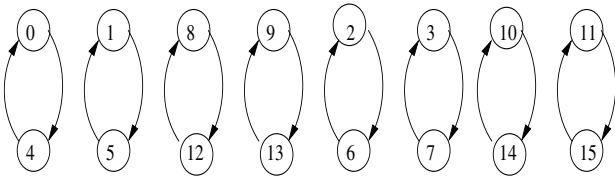


Fig. 8. State transition diagram of $\langle 204, 51, 204, 204 \rangle$

Example 3. State transitions of a 4-bit CA $\langle 204, 51, 204, 204 \rangle$ is shown in Figure 8. As the rule 204 belongs to group 8 and maintains Property 1 for each RMT, it has the highest tendency, among the 256 rules, to form single length cycle attractors in an n-cell CA. However, the presence of rule 51 dictates the shape of attractors.

Observation 6. All rules of group 1 form multi-length cycle attractors while designing uniform CA. If non-uniform CA is constructed with the rules from group 1, there is rare chance of getting single length cycle attractors.

Example 4. Let us consider the 3-cell non-uniform CA of Figure 9. Its rules are taken from group1. The state transition graph contains multi-length cycle attractors.

Observation 7. In group 2, the uniform CA designed with rule 34/48 only forms single length cycle attractors (Figure 10) -that is, SACA.

Observation 8. The uniform CA designed with the rules of group 3 forms single length cycle attractors only. Some of those - 2, 16, 32, 42, 56, 98, 112, 162, and 176 form SACA.

Example 5. Let us consider the 4-bit CA $\langle 16, 16, 16, 16 \rangle$ of Figure 11. Its state transition diagram contains only single length cycle attractor.

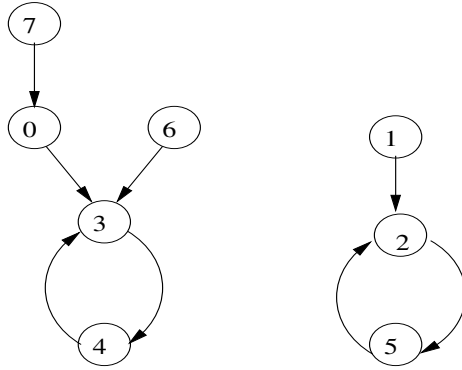


Fig. 9. State transition diagram of $\langle 50, 115, 179 \rangle$

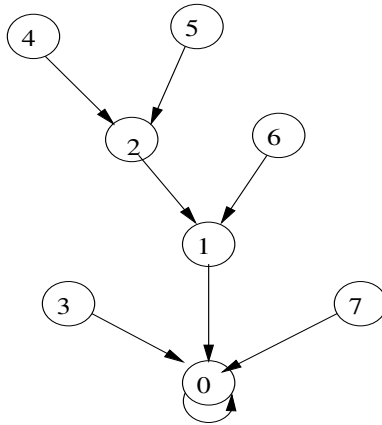


Fig. 10. State transition diagram of $\langle 48, 48, 48 \rangle$

3.2 Identification of CA Rules for SACA

The earlier subsection observed that to form an SACA, the CA rule should follow Property 1. However, a rule that maintains Property 1 for all its RMTs can't form an SACA (e.g. rule 204).

Property 2: For an uniform SACA, the CA rule must deny Property 1 for some RMTs.

Example 6. The rule 48 of group 2 denies Property 1 for 6 RMTs and follows for only 2 RMTs. On the other hand, rule 192 of group 6 denies Property 1 for only 2 RMTs and follows for 6 RMTs. Each of these rules forms SACA for all lengths. Table 3 displays the rules that form SACA of any arbitrary length.

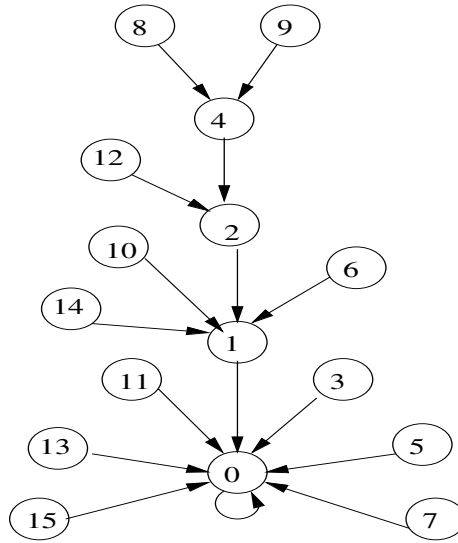


Fig. 11. State transition diagram of $\langle 16, 16, 16, 16 \rangle$

Table 3. CA rules for SACA

group	Rule
2	34, 48
3	2, 16, 32, 42, 56, 98, 112, 162, 176
4	0, 10, 15, 24, 40, 66, 80, 84, 96, 130, 144, 160, 170, 184, 226, 240, 255
5	8, 64, 128, 138, 143, 152, 168, 194, 208, 213, 224
6	136, 192

4 Faulty Sensor Node Detection Scheme

In a sensor network, hundreds of sensor nodes are deployed randomly [16]. The nodes have the ability to sense the environment, can perform some computations, and communicate with the neighbors. However, the sensor nodes are constrained by the limited battery power since recharging is very difficult and sometimes unfeasible. Therefore, any scheme devised for a wireless sensor network (WSN) should be energy efficient.

In WSN, some of the sensor nodes may become faulty due to their low battery power or some other physical defects [17]. These may send erroneous data and consume some bandwidth as well as incur extra computational overhead. Therefore, identification of faults in sensor network nodes is a necessity. In this work, we propose an SACA based scheme that can efficiently identifies the faults in network nodes without consuming much computational overhead and bandwidth.

The scheme: For a network with n nodes, we employ an n -cell CA and an n -bit all 1s seed. We assume that each sensor node N_i can check its status and whenever communicates with the base station/coordinator, it faithfully sends its status.

Let us assume that status 0 of a node implies it is fault free and 1 is for faulty. That is, if the network is fault free, each node sends ‘0’ to the coordinator; otherwise coordinator assumes ‘1’. For each node, the coordinator assigns a cell of the n -cell CA. The information received from node N_i is then used to set the rule of the i^{th} CA cell. The ‘0’ is encoded as the rule 192 (Figure 12). As $M_i=0$, $NS_i = x_{i-1} x_i$, NS_i is equivalent to rule 192. When there is a fault in node N_i , $M_i=1$, $NS_i = x'_{i-1} + x_i$. Therefore, rule 207 is set for the i^{th} CA cell (Figure 12).

Once the message is received at the coordinator, the CA is run for t -steps initialized with all 1s seed. The fault free CA is a uniform SACA constructed with rule 192 and it reaches the attractor state 0 (Figure 6). The following property of CA justifies the application of rule 207 for Cell i when the node N_i is faulty.

Property 3: If a uniform SACA with rule R_o is hybridized by a cell rule R_h , it can generate new attractors only if the set of RMTs of R_o for which Property 1 are denied is not a subset of the set of RMTs of R_h , for which also the property 1 are denied.

The set of RMTs of rule 192 (R_o) for which Property 1 are denied is 2,3 and the similar set for rule 207 is 0,1. Therefore, rule 192 and 207 follow Property 3. That is, the CA resulted from an SACA (with rule 192) due to faults at single or multiple nodes is a non-uniform (hybrid) CA (hybridized with rule 207). It generates multiple attractors and settles to an attractor with LSB as 1 (Figure 13) when initialized with all 1s seed. That is, by sensing the LSB of the CA the coordinator can detect faults in its nodes.

Once the fault in a node is detected, the faulty nodes can be diagnosed by running the CA with all 0s seed. The position of 1s in the attractors (4 of Figure 13(a), 6 of Figure 13(b), and 5 of Figure 13(c)) denote the faulty node positions.

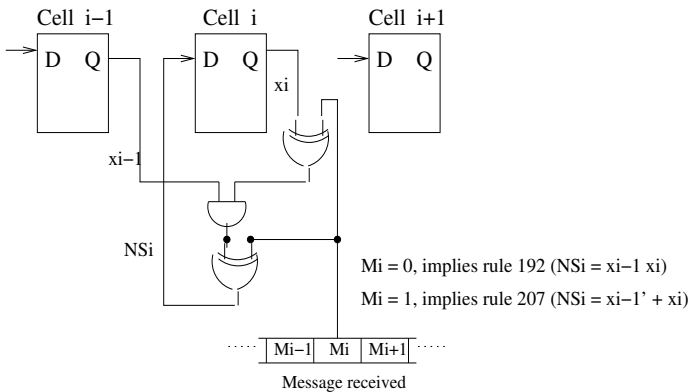


Fig. 12. CA rule setting for fault free and faulty nodes in a WSN

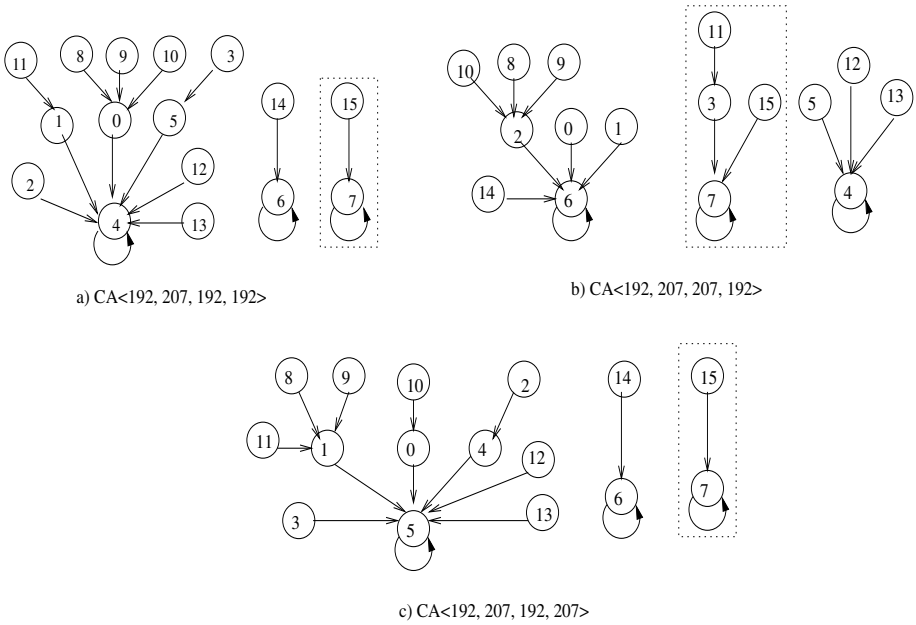


Fig. 13. State transitions of CA due to faults

5 Conclusion

The work characterizes a single attractor CA (SACA) through classification of all the 256 rules of 3-neighborhood CA. The characterization enables design of a CA based scheme for detection of faulty nodes in a wireless sensor network. It identifies faults even in multiple nodes without major computation overhead, that is desired for power constrained wireless sensor network.

References

1. Wolfram, S.: Cellular Automata and Complexity — Collected Papers. Addison Wesley, Reading (1994)
2. Martin, O., Odlyzko, A.M., Wolfram, S.: Algebraic Properties of Cellular Automata. *Comm. Math. Phys.* 93, 219–258 (1984)
3. Wolfram, S.: Statistical mechanics of cellular automata. *Rev. Mod. Phys.* 55(3), 601–644 (1983)
4. Wolfram, S.: Universality and Complexity in cellular automata. *Physica D* 10, 1–35 (1984)
5. Wolfram, S.: Undecidability and intractability in theoretical physics. *Phys. Rev. Lett.* 54, 735–738 (1985)
6. Wolfram, S.: Random sequence generation by cellular automata. *Advances in Applied Mathematics*, 123–169 (1986)

7. Pal Chaudhuri, P., Roy Chowdhury, D., Nandi, S., Chatterjee, S.: Additive Cellular Automata – Theory and Applications, vol. 1. IEEE Computer Society Press, California (1997) ISBN 0-8186-7717-1
8. Wuensche, A., Lesser, M.J.: The Global Dynamics of Cellular Automata. In: Santa Fe Institute Studies in the Science of Complexity. Addison Wesley, Reading (1992)
9. Ganguly, N.: Cellular Automata Evolution: Theory and Applications in Pattern Recognition and Classification. PhD thesis, Bengal Engineering College (a Deemed University), India (2004)
10. Maji, P.: Cellular Automata Evolution For Pattern Recognition. PhD thesis, Bengal Engineering College (a Deemed University), India (2004)
11. McIntosh, H.V.: Linear cellular automata via de bruijn diagrams (May 1991) (preprint)
12. Sutner, K.: De bruijn graphs and linear cellular automata. *Complex Systems* 5(1), 19–30 (1991)
13. Mukherjee, M., Ganguly, N., Pal Chaudhuri, P.: Cellular Automata based authentication (CAA). In: Bandini, S., Chopard, B., Tomassini, M. (eds.) ACRI 2002. LNCS, vol. 2493, pp. 259–269. Springer, Heidelberg (2002)
14. Mukhopadhyay, D., Joshi, P., RoyChowdhury, D.: An efficient design of cellular automata based cryptographically robust one-way function. In: VLSID (2007)
15. Cho, S.-J., Choi, U.-S., Hwang, Y.-H., Kim, H.-D., Choi, H.-H.: Behaviors of Single Attractor Cellular Automata over Galois Field $GF(2^p)$. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 232–237. Springer, Heidelberg (2006)
16. Megerian, S., Polkonjak, M.: Wireless Sensor Network. In: Proakis, J.G. (ed.) Wiley Encyclopedia of Telecommunications (December 2002)
17. Chen, J., Kher, S., Somani, A.: Distributed fault detection of wireless sensor networks. In: Workshop DIWANS, pp. 65–72 (2006)

Cellular Automata Applied in Remote Sensing to Implement Contextual Pseudo-fuzzy Classification

Moisés Espínola¹, Rosa Ayala¹, Saturnino Leguizamón²,
Luis Iribarne¹, and Massimo Menenti³

¹ Applied Computing Group, University of Almería, Spain

² Regional Faculty, National Technological University, Mendoza, Argentina

³ Aerospace Engineering Optical and Laser Remote Sensing, TUDelft, Netherlands

Abstract. Nowadays, remote sensing is used in many environmental applications, helping to solve and improve the social problems derived from them. Examples of remotely sensed applications include soil quality studies, water resources searching, environmental protection or meteorology simulations. The classification algorithms are one of the most important techniques used in remote sensing that help developers to interpret the information contained in the satellite images. At present, there are several classification processes, i.e., maximum likelihood, parralepiped or minimum distance classifier. In this paper we investigate a new satellite image classification Algorithm based on Cellular Automata (ACA), a technique usually used by researchers on complex systems. There are not previous works related to satellite image classification with cellular automata. This new kind of satellite image classifier, that improves the results obtained by classical algorithms in several aspects, has been validated and experimented in the SOLERES framework.

1 Introduction

Remote sensing is the most relevant science that allows us the acquisition of information about the surface of the land and environmental information values without having actual contact with the area being observed [14]. This science can be used in many environmental applications, helping to solve and improve the problems derived from them. Examples of remotely sensed applications include soil quality studies, water resources searching, environmental protection or meteorology simulations, among others.

The classification algorithms are one of the most important techniques used in remote sensing that help developers to interpret the information contained in the satellite images. The aim of satellite images classification is to divide image pixels into discrete classes (spectral classes). The resulting classified image is essentially a thematic map of the original image [15]. These algorithms have reached a great advance in the last years. The analysts use the classification algorithms to interpret the information contained in the satellite images. In the literature, there are different procedures to classify satellite images.

In spite of the great number of classifiers that exist, there are several researchers studying new classification methods because there is not a 100% efficient classifier. In this paper we propose a new procedure for the classification of satellite images. The new classification Algorithm based on Cellular Automata (ACA) uses this technique for the assignment of the satellite image pixels to the different spectral classes.

The rest of the paper is structured as follows. Section 2 describes the basic features of the spectral and contextual image classification algorithms. Section 3 describes the basic aspects of cellular automata and the classical applications in remote sensing. In section 4 we focus in the use of cellular automata to classify satellite images (ACA algorithm). Section 5 shows the results obtained with the application of ACA in the SOLERES framework. Finally, in section 6, we finish the paper exposing future work.

2 Spectral and Contextual Classification of Satellite Images

Common classification procedures can be broken down into two divisions based on the method used: supervised and unsupervised classification, whose classification methods are based on the spectral properties of the satellite image pixels. The use of supervised or unsupervised procedures depends of the analyst knowledge about the zone to study [1].

In an unsupervised classification algorithm, the analyst only specifies the number of classes, and the algorithm groups the satellite image pixels based solely on the numerical information in the data. In these algorithms, the analyst has not to know the zone to study. There are many unsupervised classification algorithms, like the Isodata algorithm, K-means, Leader, MaxiMin or Neural Model unsupervised.

In a supervised classification, the analyst selects samples of the different elements to identify the pixels in the image. In this method the analyst knowledge of the study area determines the quality of the training set. Then, the computer uses an algorithm to compare each pixel in the image to these signatures. The pixels are labelled as the class most closely resembles digitally [2]. There are several types of statistics based supervised classification algorithms. Some of the more popular ones are parallelepiped, minimum distance, maximum likelihood, Fuzzy supervised, neural model and Mahalanobis distance, among others.

These spectral supervised and unsupervised classification algorithms works well in non-noisy images and if the spectral properties of the pixels determine the classes sufficiently well. However, if noise or substantial variations in class pixel properties are present, the resulting image classification may have many small (often one-pixel) regions which are misclassified. Several standard approaches can be applied to avoid this misclassification, like using contextual information in addition to spectral data. There are several contextual classification algorithms that use mean values, variances or texture description from a pixel neighbourhood to improve that pixel spectral classification.

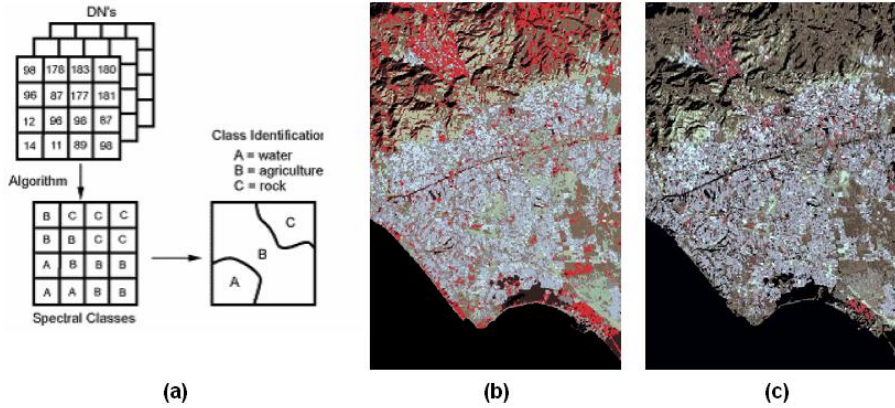


Fig. 1. (a) General classification process description, (b) Supervised classification results and (c) Unsupervised classification results

3 Classical Applications of Cellular Automata in Remote Sensing

A cellular automaton consists of a grid of cells distributed normally in a matrix form that has the following basic features:

- **States:** each cell can take an integer value that corresponds to its current state. There is a finite set of states.
- **Neighbourhood:** a set of cells that interact with the current one.
- **Transition function f :** takes as input arguments the cell and neighbourhood states, and returns the new state of the current cell.
- **Rules:** the transition function f uses a set of rules that specify how the states of the cells change.
- **Iterations:** the transition function f is applied to each cell of the grid across several iterations.

When we work with satellite images, we consider each pixel of the image as a cell of the cellular automaton and we normally take the 8 around pixels as neighbourhood (Moore Neighborhood), although we can take the 4 around pixels (von Neumann Neighborhood) or even the 24 around pixels (Extended Moore Neighborhood).

The changes in cells states occur in discrete time form. In each iteration the whole cells are checked and rules are applied through the transition function f to each cell taking into account the around neighbourhood to change its state. Therefore cellular automata have an evolution process because the cells are always changing their states across the different iterations. From this point of view, in recent years cellular automata have become a powerful tool applied in remote sensing especially to simulate satellite images processes.

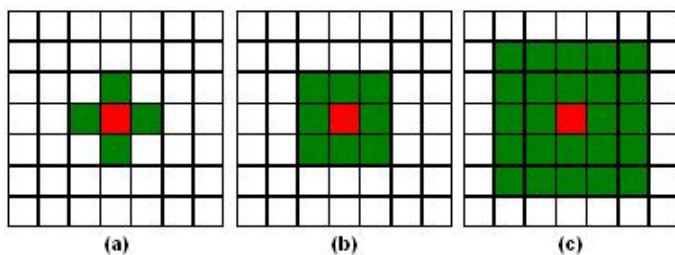


Fig. 2. (a) Von Neumann Neighborhood, (b) Moore Neighborhood and (c) Extended Moore Neighborhood

Cellular automata have been used in applications that experiment a time evolution like environmental simulations, complex social phenomena modelling, images treatment in artificial vision, cryptography of digital information and artificial intelligence in mathematical games.

Cellular automata have been widely used for environmental simulations like modelling land features dynamics [4], simulating snow-cover dynamics [5], modelling vegetation systems dynamics [3], detecting vibrio cholerae by indirect measurement [6], simulating forest fire spread [11] [12], modelling biocomplexity of deforestation process [10] and simulating land use dynamics [8]. Besides we can find cellular automata used to model complex social phenomena like studying plant population spread in controllable systems [16]. Cellular automata have been also applied in image enhancement (noise-reduction filters) and edges detection [13].

So far cellular automata have been applied on satellite images mainly to simulate processes. In the next section we propose an important and novel alternative: cellular automata applied in remote sensing to implement contextual classification algorithms of satellite images.

4 Classification of Satellite Images with Cellular Automata (ACA)

The application of cellular automata in satellite image classification processes is a new field of investigation. In this paper we propose a methodology to implement a new satellite image classification Algorithm with Cellular Automata (ACA) that classifies the pixels based on mixed spectral and contextual information, and thus improves the classification results obtained by another classical classification algorithms. This kind of Information System is designed to solve environmental problems, facilitating risk analysis and improving environmental stewardship. There are not previous works related to satellite image classification with cellular automata, only works about cellular automata applied in post-classification processes [9] and pattern classification [7].

ACA has been implemented with Visual C++ and Erdas Imagine 9.1 Toolkit, and is a new classification algorithm based on a multistate cellular automaton

that allows the user to enter new states and rules to the cellular automata in order to customize as much as possible the satellite images classification process. In order to implement ACA we must take into account the following correspondences between a cellular automaton and the basic elements of a generic process of satellite image classification:

- Each cell of the grid corresponds to a pixel of the image.
- Each state of cellular automaton will represent a different class of the final classification.
- The neighbourhood of each cell will consist of the 8 nearest cells (Moore neighbourhood).
- The transition function f must correctly classify each pixel of the image based on the features of the current cell and its neighbourhood, using mixed spectral and contextual data.

In order to customize the classification process, the satellite image expert analyst has to set the desired behavior of ACA through introducing the states and rules of the cellular automaton that defines the results wanted. For example, we have implemented a version of ACA that try to get 3 objectives. Primarily to improve the results obtained by supervised classification classical algorithms (eg minimum distance) using contextual information. Secondly to get a pseudo-fuzzy classification based on spectral proximity hierarchies, where in each iteration of cellular automata only those pixels that are within a spectral distance of the center of its class are classified (this distance is increased in each iteration). And thirdly, to obtain a detailed list of the noisy and uncertain pixels, and classes edges detection. So we have assigned 3 states for each of the cells: [class][quality][type], where each state can take the following values:

- [class]=training set classes, noiseClass (noisy pixels) or emptyClass (pixels not classified yet).
- [quality]= $1.\text{numIterations}$ (number of iterations of CA)
- [type]= focus (not border pixels), edge (border pixels), uncertain (caotic pixels) and noisy (noise detection).

This version of ACA is based on the minimum distance supervised classifier, so the cellular automaton uses the results of minimum distance spectral classification to apply its rules. The transition function f take into account the following inputs to apply the cellular automaton rules:

- Neighbourhood states: states of actual pixel neighbourhood.
- Spectral classification classes of minimum distance algorithm: classes set of actual pixel (maybe one class, or several classes if is an uncertain pixel that is near two or more classes).
- CA iteration: actual iteration of CA.

The cellular automaton rules that gets the 3 objectives that we have described above are the following:

- If the number of spectral classification classes is 1, and the neighbourhood class states are *emptyClass* or the same as actual pixel:
[class][quality][type] = spectralClass, CAiteration, focus
- If the number of spectral classification classes is 1, and the neighbourhood class states are different than actual pixel class:
[class][quality][type] = spectralClass, CAiteration, edge
- If the number of spectral classification classes is 1 and the spectralClass is *noiseClass*:
[class][quality][type] = majority class of the neighbourhood, CAiteration, noisy
- If the number of spectral classification classes is bigger than 1:
[class][quality][type] = majority class of the neighbourhood among the dubious classes, CAiteration, uncertain

In the next section we show the results obtained with this particular version of the ACA algorithm.

5 Results and Conclusions

In this section we analyze the results obtained with this version of ACA algorithm. Tests have been carried out on a multispectral Landsat image with 7 layers, with a total resolution of 301x301 pixels (90,601 total pixels). The spatial resolution of each pixel is 30x30 meters.

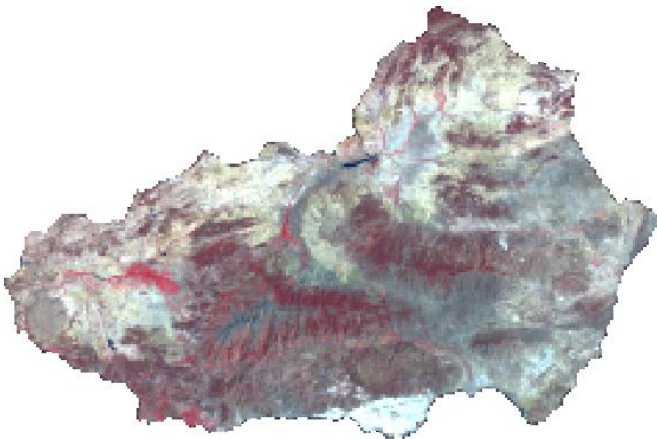


Fig. 3. Complete satellite image of Almería and Granada provinces (Spain)

The image corresponds to a region of the provinces of Almería and Granada (Spain), and the image pixels were classified in a total of 8 classes. This is a region with a significant percentage of uncertain pixels and a minimum percentage of noisy pixels. With this version of ACA we have achieved the following objectives:

a) **Improving the quality of the pixels classification using contextual information.**

ACA improves the results obtained by another supervised classification algorithms, because in the classification process of each image pixel it uses the around pixels as neighbourhood in the transition function f , and this relationship among the image pixels offers an optimal final classification. In this experiment we compared the classification results obtained from the classical minimum distance algorithm with ACA based on minimum distance. To compare these algorithms have calculated the confusion matrix between the two classified images and an image from either ranked fieldwork. The comparison can be seen in the following two tables:

Table 1. Confusion matrix of the minimum distance algorithm

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8
Class 1	0	0	0	0	0	0	0	0
Class 2	789	6080	355	0	0	0	0	0
Class 3	1247	0	9447	532	0	0	0	0
Class 4	1547	0	2	11998	242	0	0	0
Class 5	1555	0	0	52	12827	3	27	0
Class 6	1027	0	0	47	281	8330	35	1
Class 7	1513	0	0	0	250	12	13050	0
Class 8	1396	0	0	0	0	66	381	11242

Table 2. Confusion matrix of the ACA algorithm

	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8
Class 1	0	0	0	0	0	0	0	0
Class 2	123	6326	611	54	62	11	18	14
Class 3	179	0	9648	963	240	74	73	37
Class 4	205	0	5	12282	777	252	174	73
Class 5	194	0	1	58	13254	88	713	135
Class 6	123	0	0	47	291	8699	161	330
Class 7	130	0	0	1	254	16	13724	661
Class 8	217	0	0	0	0	68	407	11966

The number of well classified pixels in each algorithm is obtained by adding the values in the main diagonal of the table. In the case of the minimum distance algorithm there are a total of 72.974 well classified pixels (80% well classified), and in the ACA algorithm there are a total of 75.899 well classified pixels (84% well classified). Therefore the quality of the final classification has improved by 4%, also we have grouped in a single algorithm a preclassification process(noise reduction), the proper classification process and a postclassification process(improving uncertain pixels).

b) **Obtaining a pseudo-fuzzy classification based on spectral proximity hierarchies in feature space.**

With ACA we can obtain a hierarchical classification based on spectral proximity in feature space, so that in each iteration of cellular automaton only those pixels in the image that are within a distance from the center of its class are classified, and this distance is increasing at each iteration. Thus, the pixels classified in a particular iteration are more reliable than those that fall in the next iteration, and so on.

This method of classification has similar behavior to the fuzzy classification, although not exactly alike, so we have called pseudo-fuzzy. In the Figure 4 we show a sequence of images where we can see the results of our image classification by dividing the process in 6 iterations of the cellular automaton.

The assignment of colors to each class is set to gray scale, so that black pixels are those that have not yet been classified. As you can see in the first iteration, the number of classified pixels are quite small. These pixels are those that are closer from the standpoint of the average spectral classes, and therefore are more reliable. Some of these pixels even belong to the training set chosen by the expert. In the next iterations the threshold distance to the average of each class will increase, so that others pixels that are spectrally farther in its class will be classified. In the last iteration, the uncertain and noisy pixels are classified, which are often the most problematic. Thanks to these results, experts can detect visually much more reliable problematic

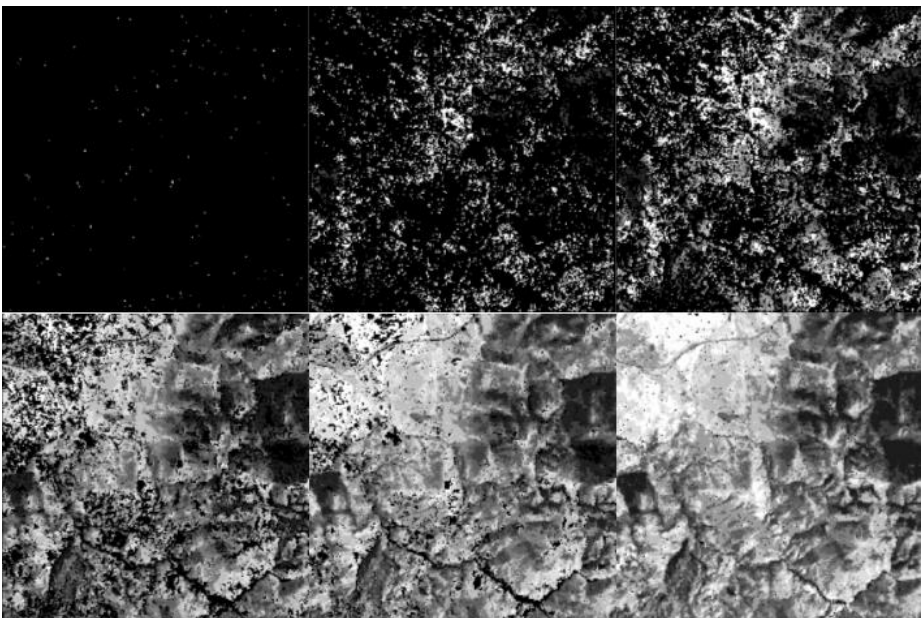


Fig. 4. Pseudo-fuzzy classification with 6 iterations of cellular automaton

pixels in the classification process, valuable information in the process of satellite images analysis.

c) **Boundary, uncertain and noisy pixel detection.**

In addition, the ACA algorithm also provides the expert with a list of border pixels of each class represented in the image as well as uncertain and noisy pixels, in order to have more additional information related to the classification process to improve the subsequent analysis of results obtained. Thus, the ACA algorithm incorporates aspects of pre-sorting tasks (detection and elimination of image noise), classification (enhanced in our case) and postclasificacin (correction uncertain pixel) satellite imagery.

6 Future Work

Some possible future work are shown below:

- a) Implementing new versions of the ACA algorithm based on new states and rules of cellular automaton to further customize the classification process.
- b) Using software agents to reduce the computational cost, touring various regions of the image in parallel.
- c) Creating a Erdas Imagine plugin that allows a custom classification based on cellular automata.

Acknowledgments

This work has been partially supported by the EU (FEDER) and the Spanish MEC under grant of the project I+D TIN2007-61497 “Soleres. A Spatio-Temporal Environmental Management System based on Neural-Networks, Agents and Software Components”.

References

1. Ayala, R., Menenti, M., Girolana, D.: Evaluation methodology for classification process of digital images Igarss 2002. In: IEEE Int. Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, pp. 3363–3365 (2002)
2. Ayala, R., Becerra, A., Flores, I.M., Bienvenido, J.F., Díaz, J.R.: Evaluation of greenhouse covered extensions and required resources with satellite images and GIS. Almería’s case. In: Second European Conference of the European Federation for Information Technology in Agriculture, Food and the Environment, Bonn, Germany, pp. 27–30 (1999)
3. Balzter, H., Braun, P., Kühler, W.: Cellular automata models for vegetation dynamics. *Ecological Modelling* 107, 113–125 (1998)
4. Leguizamón, S.: Modeling land features dynamics by using cellular automata techniques. In: Proceedings of the ISPR Technical Comision 7 Mid-Term Symposium “From pixels to Processes”, Enschede, The Netherlands, pp. 497–501 (2006)

5. Leguizamón, S.: Simulation of snow-cover dynamics using the cellular automata approach. In: Proceedings of the 8th International Symposium on High Mountain Remote Sensing Cartography, La Paz, Bolivia, pp. 87–91 (2005)
6. Lobitz, B., Beck, L., Huq, A., Woods, B., Fuchs, G., Faruque, A., Colwell, R.: Climate and infectious disease: use of remote sensing for detection of *Vibrio cholerae* by indirect measurement. Proceedings of the National Academic of Sciences of the USA 97(4), 1438–1443 (2000)
7. Maji, P., Shaw, C., Ganguly, N., Sikdar, B., Chaudhuri, P.: Theory and application of cellular automata for pattern classification. *Fundamenta Informaticae* 58(3-4), 321–354 (2003)
8. Messina, J., Walsh, S.: Simulating land use and land cover dynamics in the ecuadorian Amazon through cellular automata approaches and an integrated GIS. In: Open Meeting of the Human Dimensions of Global Environmental Change Research Community in Rio de Janeiro, Brazil, pp. 6–8 (2001)
9. Mojaradi, B., Lucas, C., Varshosaz, M.: Using learning cellular automata for post classification satellite imagery. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences* 35(4), 991–995 (2004)
10. Moreno, N., Quintero, R., Ablan, M., Barros, R., Dávila, J., Ramírez, H., Tonella, G., Acevedo, M.: Biocomplexity of deforestation in the Caparo tropical forest reserve in Venezuela: an integrated multi-agent and cellular automata model. *Environmental Modelling and Software* 22, 664–673 (2007)
11. Muzy, A., Innocenti, E., Aiello, A., Santucci, J.F., Santonio, P.A., Hill, D.: Modelling and simulation of ecological propagation processes: application to fire spread. *Environmental Modelling and Software* 20, 827–842 (2005)
12. Muzy, A., Innocenti, E., Aiello, A., Santucci, J.F., Santonio, P.A., Hill, D.: Dynamic structure cellular automata in a fire spreading application. In: First International Conference on Informatics in Control, Automation and Robotics, Setubal, Portugal, pp. 143–151 (2004)
13. Popovici, A., Popovici, D.: Cellular automata in image processing. In: Proceedings of the 15th International Symposium on the Mathematical Theory of Networks and Systems, 6p. (2002)
14. Rees, W.G.: Physical principles of remote sensing, 2nd edn. Cambridge University Press, Cambridge (2001)
15. Schowengerdt, R.A.: Techniques for image processing and classification in remote sensing. Academic Press, London (1985)
16. Wang, J., Kropff, M., Lammert, B., Christensen, S., Hansen, P.: Using CA model to obtain insight into mechanism of plant population spread in a controllable system: annual weeds as an example. *Ecological Modelling* 166, 277–286 (2003)

Impact of Coupling of Distributed Denial of Service Attack with Routing on Throughput of Packet Switching Network

Anna T. Lawniczak^{1,2}, Hao Wu², and Bruno Di Stefano^{2,3}

¹ Department of Mathematics and Statistics, University of Guelph,
Guelph, Ontario N1G 2W1, Canada

² The Fields Institute for Reserach in Mathematical Sciences, 222 College Street,
Toronto, Ontario M5T 3J1, Canada

³ Nuptek Systems Ltd., Toronto, Ontario M5R 3M6, Canada

alawnicz@uoguelph.ca, bruno.distefano@nupteksystems.com,
kamanwu@gmail.com

Abstract. We study the effects of coupling of the Distributed Denial of Service (DDoS) attack with routing on a packet switching network (PSN) performance measured by throughput. We conduct our study using PSN model that it is an abstraction of the Network Layer of the 7-Layer ISO OSI Reference Model. Our study demonstrates that even a very “weak” DDoS attack on a network using static routing causes degradation of the network throughput. The values of the throughput almost immediately decrease with each onset of a DDoS attack and they decrease with the increase of the number of attackers. However, this is not the case when the network uses an adaptive routing instead. We consider two different types of adaptive routings and our study shows that the adaptive routings have ability to process efficiently extra packet traffic generated by DDoS attacks without compromising the network throughput when the total amount of the incoming packet traffic, i.e. the regular one and the one coming from an attack, is lower than the one corresponding to the critical source load value.

Keywords: packet switching network, denial of service attack, throughput.

1 Introduction

The purpose of Denial of Service Attacks (DDoS Attacks) is to make a computer resource inaccessible to its legitimate users. In DDoS, attacks are “distributed” because the attacker gains control of a huge number of independently owned and geographically distributed computers, called “zombies”, and almost always controls them in a concealed way without any knowledge of their legitimate owners. The attacker carries on his/her actions by means of multiple “zombies”, located at various network nodes. Thus, the DDoS attack is a network attack taking advantage of asymmetry between network-wide resources and local capacity of the target (victim) machine to process incoming packet traffic. In a DDoS attack the victim machine and its neighbouring nodes may become quickly saturated with buildup of intended congestion, such that

they cannot respond to legitimate traffic any longer, [1] “Ping flood” is the type of DDoS attack directing a huge number of “ping” requests to the target victim of the attack. This type of attack exploits the “Internet Control Message Protocol” (ICMP). “Ping is a computer network tool used to test whether a particular host is reachable across an IP network”, [2]. By issuing a huge number of ping ‘echo requests’ from a very large number of “zombies” spread all over the network, it is possible to cripple the target victim and make it unable to conduct any network activity other than answering the ping ‘echo requests’ and, eventually, rendering it so overloaded that it will come to a standstill. Examples of “ping floods” attacks are DDoS attacks of the “Estonian Cyberwar” in 2007, see [3] and [4], the Mafiaboy attacks of February 2000 against Amazon, eBay that caused millions of dollars damage and which are often quoted by computer experts, see [5] and [6].

The entropy based detection of DDoS attacks of “ping floods” type was discussed in [7], [8] and [9]. Here, we investigate the impact of the coupling of DDoS attack of “ping floods” type with network routing on network performance in delivering packets to their destinations that is measured by *throughput* being an aggregate measure of network performance. We conduct our study using PSN model that is an abstraction of the Network Layer of the 7-Layer ISO OSI Reference Model [10]. Our study demonstrates that even a very “weak” DDoS attack on a network using static routing causes degradation of the network *throughput*. The values of the *throughput* almost immediately decrease with each onset of a DDoS attack and they decrease with the increase of the number of attackers. However, this is not the case when the network uses an adaptive routing instead. We consider two different types of adaptive routing and our study shows that the adaptive routings have ability to process efficiently extra packet traffic generated by DDoS attacks without compromising the network *throughput* when the total amount of the incoming packet traffic, i.e. the regular one and the one coming from an attack, is lower than the total amount of traffic corresponding to the critical source load value.

The paper is organized as follows. First, we briefly describe the abstraction of the PSN model that we use for our research [11] and [12], its C++ simulator, Netzwerk, [13] and [14], and explain how they have been customized to model “ping” type DDoS attacks in our simulation experiments. Next, we introduce definitions of some network performance indicators, e.g., *throughput*, and describe the types of simulation experiments that we perform. Finally, we present selected simulation results and our conclusions.

2 PSN and DDoS Attack Models Description

To study impact of DDoS attacks on PSN performance indicators, e.g., *throughput*, we customized our PSN model described in details in [11] and [12]. The PSN model is an abstraction of the Network Layer of the 7-Layer ISO OSI Reference Model [10]. Our PSN model is concerned primarily with packets and their routings; it is scalable, distributed in space, and time discrete. It avoids the overhead of protocol details present in many PSN simulators designed with different aims in mind than study of macroscopic network-wide dynamics of packet traffic and aggregate measures of network performance.

A PSN connection topology is represented by a weighted directed multigraph \mathbf{L} where each node/router corresponds to a vertex and each communication link is represented by a pair of parallel edges oriented in opposite directions. In each PSN model setup each cost of transmission of a packet along a link (an edge) is computed using the same type of *edge cost function* (*ecf*) that is either the *ecf* called *ONE* (*ONE*), or *QueueSize* (*QS*), or *QueueSizePlusOne* (*QSPO*). The *ecf ONE* assigns a value of “one” to all edges in the lattice \mathbf{L} . This results in a static routing since this value does not change during the course of a simulation. The *ecf QS* assigns to each edge in the lattice \mathbf{L} a value equal to the length of the outgoing queue at the node from which the edge originates. The *ecf QSPO* assigns a value that is the sum of a constant “one” plus the length of the outgoing queue at the node from which the edge originates. The routing decisions made using *ecf QS* or *QSPO* result in *adaptive* or *dynamic routing* because they rely on the current state of the network simulation and the packets are routed avoiding congested nodes during the PSN model simulation. In our PSN model, each packet is transmitted via routers from its source to its destination according to the routing decisions made independently at each router and based on a *minimum least-cost criterion* of selecting a *shortest path* from a packet current node to its destination. Thus, if the PSN model is setup with *ecf ONE* then the routing is the *minimum hop routing* (*minimum route distance*) and if it is setup with *ecf QS* or *QSPO* then it is the *minimum length routing*. It is important to notice, that in the case of PSN model setup with *ecf QS* or *QSPO*, because these costs are dynamic, each packet is forwarded from its current node to the next one that belongs to a least cost shortest path from the packet current node to its destination at this time. The PSN model uses *full-table routing*, that is, each node maintains a routing table of least path cost estimates from itself to every other node in the network. The routing tables are updated at each time step when the *ecf QS* or *QSPO* is used; see [11] and [12]. Since the values of the *ecf ONE* do not change over time the routing tables do not need to be updated for the static *ecf ONE*; see [11] and [12]. We update the routing tables using distributed routing table update algorithm [12].

In our simulations to study DDoS attacks and their impact on network performance indicators we use a version of PSN model in which each node performs the functions of *host* and *router* and maintains one incoming and one outgoing queue which is of unlimited length and operates according to a first-in, first-out policy, see [12] for other options. At each node, independently of the other nodes, packets are created randomly with probability λ called *source load*. In our PSN model all messages are restricted to one packet carrying time of creation, destination address, and number of hops taken.

In the PSN model time is discrete and we observe its state at the discrete times $k = 0, 1, 2, \dots, T$, where T is the final simulation time. In the presented simulations each PSN model setup is characterized by the selection of a network connection topology type, an *ecf* type, a type of routing table update (we use distributed routing table update), a *source load* value and final simulation time T . At time $k = 0$, the setup of the PSN model is initialized with empty queues and the routing tables are computed. The time discrete, synchronous and spatially distributed PSN model algorithm consists of the sequence of five operations advancing the simulation time from k to $k + 1$. These operations are: (1) *Update routing tables*, (2) *Create and route packets*, (3) *Process*

incoming queue, (4) *Evaluate network state*, (5) *Update simulation time*. The detailed description of this algorithm is provided in [11] and [12].

To study DDoS attacks and their impact on network performance indicators we modified the described PSN model by allowing selecting one victim computer and a user defined number of zombies either located at specified nodes or located at random. For each zombie start and end of attack time can be specified separately. As in most real life cases, zombies continue to carry on their normal jobs during an attack, i.e. they act also as sources, destinations, and routers of legitimate data transfers. However, each zombie also sends a packet to the victim at each time step of an attack simulation.

3 PSN Model and DDoS Attack Setups and Network Performance Indicators

We simulate DDoS attacks on PSN model with $L_{\square}^p(37, \text{ecf}, \lambda)$ setups, i.e. on PSN models with the network connection topology isomorphic to $L_{\square}^p(37)$ (i.e., periodic square lattice with 37 nodes in the horizontal and vertical directions) and for each of the *ecf* = *ONE*, or *QS*, or *QSPO*. The incoming traffic is generated, at the network nodes, by Bernoulli random variables with expected value λ , i.e., with λ *source load* value.

In the PSN model, for each family of network setups, which differ only in the value of the *source load* λ , values of $\lambda_{\text{sub-c}}$ for which packet traffic is congestion-free are called *sub-critical source loads*, while values $\lambda_{\text{sup-c}}$ for which traffic is congested are called *super-critical source loads*. The *critical source load* λ_c is the largest sub-critical source load. Thus, λ_c is an important network performance indicator because it is the phase transition point from free flow to congested state of a network. Details about how we estimate the *critical source load* value are provided in [12].

For the PSN model setups considered here the estimated *critical source load* (*CSL*) values are, respectively, $\lambda_c = 0.053$ for $L_{\square}^p(37, \text{ONE})$, and $\lambda_c = 0.054$ for $L_{\square}^p(37, \text{QS})$ and $L_{\square}^p(37, \text{QSPO})$.

Another very important network performance indicator is *throughput*. It measures the rate with which a network delivers packets to their destinations. Higher the rate is the more efficiently the network performs. We calculate *throughput*, $\Theta(k)$, at each time k , by taking the time-average of $N_d(k)$, i.e., of a total number of packets delivered to their destinations up to time k . Thus, $\Theta(k) = k^{-1} N_d(k)$. Fig. 1 shows *source load* dependent graphs of *throughput* at the final simulation time $k=2^{17}=131,072$ for the PSN model with the setup $L_{\square}^p(37, \text{ecf})$; the blue graph corresponds to *ecf ONE*, the green graph to *ecf QS* and the red one to *ecf QSPO*. In Fig. 1 we observe that the graphs of *throughput* attain their maximum values at $\lambda=0.054$, i.e. at λ_c for the PSN model with setup $L_{\square}^p(37, \text{QS})$ or $L_{\square}^p(37, \text{QSPO})$ and at a slightly higher value than λ_c for the PSN model with the setup $L_{\square}^p(37, \text{ONE})$. Also, we observe that the graph of *throughput* of the PSN model with the setup $L_{\square}^p(37, \text{QS})$ is almost identical to the one of the PSN model with the setup $L_{\square}^p(37, \text{QSPO})$ and that they are similar to the graph of *throughput* of the PSN model with the setup $L_{\square}^p(37, \text{ONE})$. Thus, from the *throughput* point of view, the PSN models under normal traffic conditions perform are very similarly, regardless what type of the *ecf* they use, i.e. under normal traffic conditions *throughput* are similar ones, whether the static or one of the dynamic routings is used.

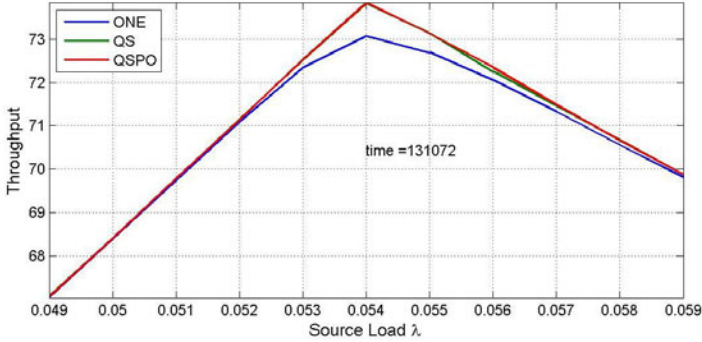


Fig. 1. Source load dependent graphs of *throughput* at the final simulation time $T=2^{17}=131,072$. The blue graph corresponds to PSN model with the setup $L^p_{\square}(37, ONE)$, the green graph to PSN model with the setup $L^p_{\square}(37, QS)$ and the red graph to PSN model with the setup $L^p_{\square}(37, QSP0)$.

However, this is not the case when the PSN model is under DDoS attack. Under anomalous traffic conditions network *throughput* may become badly affected by the coupling of DDoS attack with a routing type, see Section 4.

We carry out the simulations of DDoS attacks on PSN model setup with *source load* value $\lambda=0.040$, i.e. for *sub-critical source load* value of incoming packet traffic. At this *source load* value packet traffic of each PSN model, regardless of it setup, is in free flow state, i.e. is free of any congestion. The number of packets in transit does not increase with time and fluctuates around some constant value after an initial transient time, see [9]. Recall, that we start each simulation of a PSN model setup always with empty queues.

We start all DDoS attacks after the initial transient times, i.e. when each network operates already in its normal steady state for some time. For the presented simulation results we selected time $k_0 = 20,480$ as the start time of all the DDoS attacks. This time is much longer than the transient times of the considered PSN model setups. All the DDoS attacks last until the final simulation time, $T = 2^{17}=131,072$, being the same for all the PSN model setups. In our simulations we consider a series of separate DDoS attacks on a victim having always the same location in all the experiments. Each attack is characterized by a number of active attackers/zombies. In this series of attacks, while increasing number of zombies we maintained always the same locations of the zombies from the DDoS attacks with their lower numbers, i.e. each time we only add a new zombie to the set of the zombies from the previous attack. During an attack each active zombie at each simulation time step sends a packet to the victim. Thus, the considered DDoS attacks are abstractions of the “ping” type of DDoS attacks.

4 Impact of Coupling of DDoS Attack with Routing on PSN Throughput

In this paper we present selected simulation results illustrating impact of the coupling of DDoS attack with routing on PSN model *throughput*. Fig. 2 to Fig. 4 display time

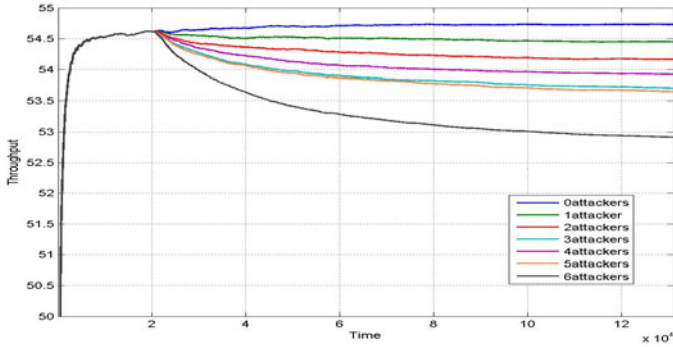


Fig. 2. Time dependent graphs of *throughput* functions of PSN model with the setup $L^p_{\square}(37, ONE, 0.040)$ being under DDoS attack characterized by the number of active zombies listed in the figure legend. Each attack starts at time $k_0 = 20,480$ and last until the final simulation time $T = 2^{17} = 131,072$.

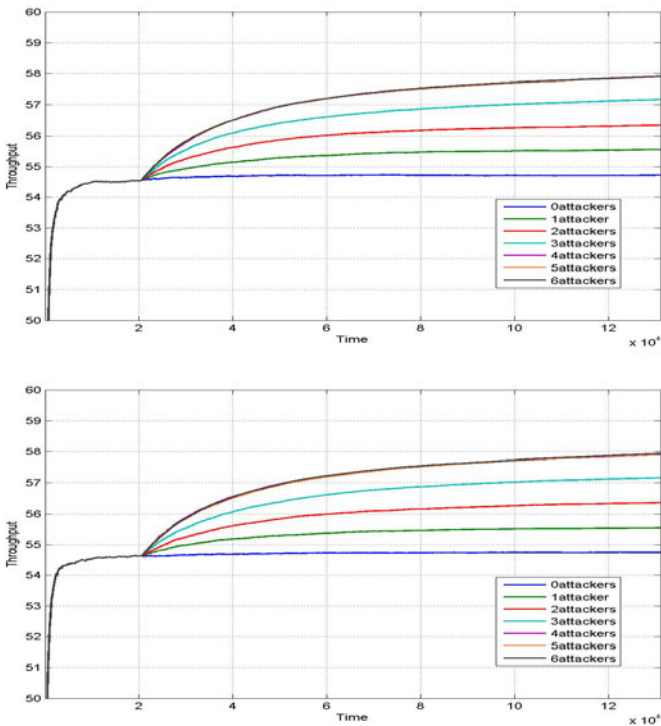


Fig. 3. Time dependent graphs of *throughput* functions of PSN model with the setup $L^p_{\square}(37, QS, 0.040)$ (top figure) and with the setup $L^p_{\square}(37, QSPO, 0.040)$ (bottom figure) being under DDoS attack characterized by the number of active zombies listed in the figure legend. Each attack starts at time $k_0 = 20,480$ and last until the final simulation time $T = 2^{17} = 131,072$.

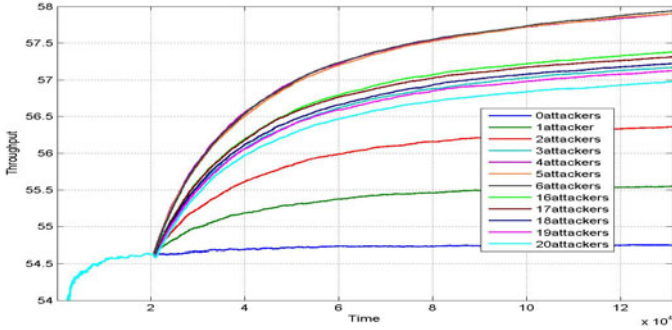


Fig. 4. Time dependent graphs of *throughput* functions of PSN model with the setup $L^p_{\square}(37, QSPO, 0.040)$ being under DDoS attack characterized by the number of active zombies listed in the figure legend. Each attack starts at time $k_0 = 20,480$ and last until the final simulation time $T = 2^{17}=131,072$.

dependent graphs of *throughput* functions of PSN model being under DDoS attacks, when the network model uses, respectively, various *ecfs*, i.e., *ONE*, *QS*, and *QSPO*. The graphs display time dependent network *throughput* functions at time intervals of 128 simulation time steps from the start of each simulation at $k=0$ to its final simulation time at $T = 2^{17}=131,072$. In Fig. 2 are displayed the graphs of *throughput* functions of PSN model with the setup $L^p_{\square}(37, ONE, 0.040)$, i.e., when PSN model uses the static routing. In Fig. 3 and Fig. 4 are displayed the graphs of *throughput* functions of PSN model when it uses dynamic/adaptive routing, i.e., for PSN model with the setup $L^p_{\square}(37, QS, 0.040)$, top plot of Fig. 3 and with the setup $L^p_{\square}(37, QSPO, 0.040)$, bottom plot of Fig. 3 and Fig. 4. In Fig. 2 and Fig. 3 are displayed the graphs of *throughput* functions when the PSN model is under DDoS attacks, respectively, with the number of active attackers/zombies varying from 0 to 6, i.e. with the number of zombies varying from 0% to about 0.44% of the total number of 1,369 nodes in the network. Fig. 4 illustrates how the impact of the coupling of the dynamic routing, using *ecf QSPO*, with DDoS attacks changes when the numbers of active attackers/zombies increase. To better illustrate this change; in Fig. 4 we display the graphs of *throughput* functions not only for the number of active attackers/zombies varying from 16 to 20, but also when their numbers vary from 0 to 6. Not displayed here, the graphs of *throughput* functions when *ecf QS* is used instead of *ecf QSPO* look very similar.

In Fig. 1 to Fig. 3 we observe that when incoming packet traffic is normal (i.e., when number of attackers is zero) the graphs of *throughput* functions behave very similarly, regardless which type of routing is used, i.e. which type of *ecf* the PSN model is using. In Fig. 2 and Fig. 3 we see that for the normal incoming packet traffic (i.e., when the number of attackers is zero) the graphs of *throughput* functions are almost constant after some initial transient times. The *throughput* constant values are almost identical ones when *ecf QS* or *ecf QSPO* is used and they differ only slightly from the *throughput* constant value when *ecf ONE* is used instead. The similar behaviours of the time dependent *throughput* functions are observed for other *sub-critical source load* values of the normal incoming packet traffic. However, in the presence of anomalous incoming packet traffic, i.e. in the presence of DDoS attacks, we observe

that the coupling of DDoS attack packet traffic with routing may have a big impact on the network performance in delivering packets to their destinations, i.e., on the network *throughput* functions, even when the *sub-critical source load* values of the normal incoming packet traffic are low. This is the case of the considered examples, as can be seen from Fig. 2 to Fig. 4.

When the PSN model uses the static routing, i.e. it uses *ecf ONE*, as soon as each DDoS attack starts the values of the *throughput* functions almost immediately decrease from the constant value of the *throughput* function corresponding to the case of the normal incoming packet traffic. In Fig. 2 we observe that with the increase of the number of attackers in DDoS attack the values of the corresponding *throughput* functions decrease and in each case they also decrease with time, i.e. with the duration of each DDoS attack. The reason for this behaviour is that the network is using a static routing. When the PSN model uses the static routing even a “weak” DDoS attack (e.g., even with 1 attacker) may cause very quickly build up of congestion along the shortest paths from an attacker or the attackers to the victim, as can be seen from Fig. 5 and Fig. 6. In these figures are displayed the sizes of packet queues at the nodes of the PSN model when a victim node is under an attack with 2 attackers and 6 attackers, respectively, in Fig. 5 and Fig. 6.

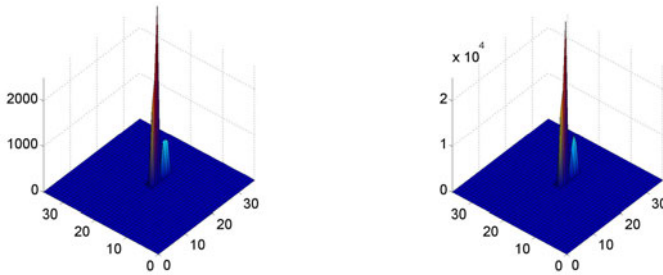


Fig. 5. Snapshot of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, ONE, 0.040)$ setup being under DDoS attack with 2 attackers at time $k = 2^{15} = 32,768$ (left plot) and at time $k = 2^{17} = 131,072$ (right plot). The maximum packet queue size excluding the zombies’ queues is 3,810 in the left plot and 34,737 in the right plot.

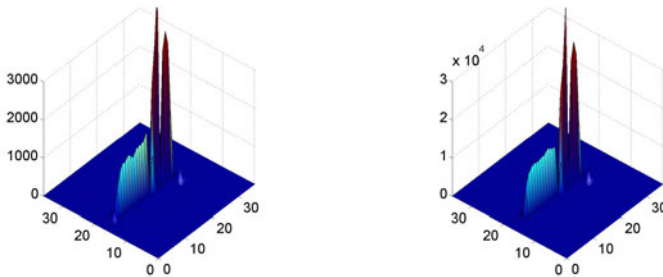


Fig. 6. Snapshot of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, ONE, 0.040)$ setup being under DDoS attack with 6 attackers at time $k = 2^{15} = 32,768$ (left plot) and at time $k = 2^{17} = 131,072$ (right plot). The maximum queue size excluding the zombies’ queues is 5,082 in the left plot and 46,106 in the right plot.

When static routing is used packets do not have an ability to avoid congested nodes on their routes from their sources to their destinations. Thus, the local congestions grow very quickly along the shortest paths from the zombies to the victim, even in the case of a small number of attackers, while the queue sizes outside these shortest paths remain more or less of the same magnitude, see Fig. 5 and Fig. 6. Since even for a “weak” DDoS attack large number of packets becomes quickly trapped in the queues, these packets are not delivered to their destinations in a timely manner. This results in decrease of the values of $N_d(k)$, i.e. a total number of packets delivered to their destinations up to time k , with increase of time and with increase of the number of attackers. Thus, ultimately this results in decrease of the values of the *throughput* functions, i.e., $\Theta(k)=k^{-1}N_d(k)$, with increase of time and with increase of the number of attackers. This happens even though the total amount of incoming packet traffic of the considered DDoS attacks (i.e., of the normal incoming packet traffic plus the one coming from a DDoS attack) is lower than the total amount of incoming packet traffic corresponding to the *critical source load* value of the PSN model with $L^p_{\square}(37, ONE)$ setup.

Let us mention that for the normal incoming packet traffic the values of *throughput* functions decrease with time after some transient times only for *super-critical source load* values but not for *sub-critical source load* values for which they fluctuate slightly around respective constant values. For the *super-critical source load* values the *throughput* functions decrease with time because the network becomes globally congested and the congestion grows with time. Thus, many packets are trapped in the queues and they are not delivered to their destinations in the timely manner.

In Fig. 3 we observe that the graphs of *throughput* functions behave very differently when the network, being under “weak” DDoS attacks, uses adaptive routing instead of using the static routing, i.e. when it uses *ecf QS* or *ecf QSPO* instead of using *ecf ONE*. Also, we notice that when PSN model uses *ecf QS* or *ecf QSPO* the qualitative and quantitative behaviours of the *throughput* functions are very similar ones, see Fig. 3. The same holds when the number of attackers is larger. Thus, the dominant effect on the behaviour of the *throughput* functions when the *ecf QSPO* is used has the dynamic cost component *QS* of the *ecf QSPO* but not the static cost component *ONE*.

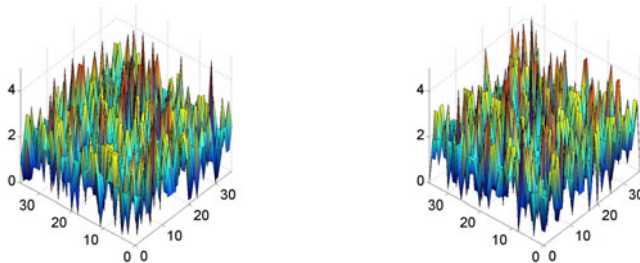


Fig. 7. Snapshot of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, QSPO, 0.040)$ setup being under DDoS attack with 2 attackers at time $k = 2^{15} = 32,768$ (left plot) and at time $k = 2^{17} = 131,072$ (right plot). The maximum packet queue size excluding the zombies’ queues is 5 in the left plot and 6 in the right plot.

In Fig. 3 and Fig. 4 we observe that for “weak” DDoS attacks, i.e. with the number of attackers smaller than 6, the values of *throughput* functions increase with time and with increase of the number of attackers when the PSN model uses *ecf QS* or *ecf QSPO*. There are several reasons responsible for such behaviour of the *throughput* functions. The most important one is that the network uses the adaptive routing. This type of routing has the ability to route packets avoiding congested nodes. Thus, the adaptive routing redistributes rather evenly the packets, i.e. those coming from the normal packet traffic and the extra ones coming from a DDoS attack, among the network nodes. This prevents the build up of local congestions in the network, except of the neighbouring nodes of the victim for stronger attacks as can be seen in Fig. 8 but is not Fig. 7. When the volume of the normal incoming packet traffic is low it may happen that for “weak” DDoS attacks the total amount of incoming packet traffic, i.e. of the normal one and the one coming from a DDoS attack, is lower than the total amount of incoming packet traffic corresponding to the *critical source load* value of the PSN model, case of Fig. 3. For such amounts of the total incoming packet traffic the congestion does not build up globally and, if it builds up at all, it builds up only at the neighbouring nodes of the victim (see Fig. 8), when the DDoS attack exceeds the victim’s capacity to absorb the incoming traffic, e.g. 4 packets at each time step in the case of considered network connection topology. In Fig. 7 are displayed snapshots of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, QSPO, 0.040)$ setup being under DDoS attack with 2 attackers at time $k = 2^{15} = 32,768$ on the left plot and at time $k = 2^{17} = 131,072$ on the right plot. On these snapshots we do not see any local congestion round the victim, as this “weak” DDoS attack does not exceed the victim’s capacity to absorb the incoming packet traffic. However, we see the build up of local congestion around the victim in Fig. 8. This figure displays snapshots of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, QSPO, 0.040)$ setup being under DDoS attack with 6 attackers at time $k = 2^{15} = 32,768$ on the left plot and at time $k = 2^{17} = 131,072$ on the right plot of Fig. 8. However, this local congestion does not spread out globally. Thus, the majority of the packets are delivered to their destinations in a timely manner even when their amount is increased due to a “weak” DDoS attack on the network. Hence, a total number of packets delivered to their destinations up to time k , i.e. $N_d(k)$, increases

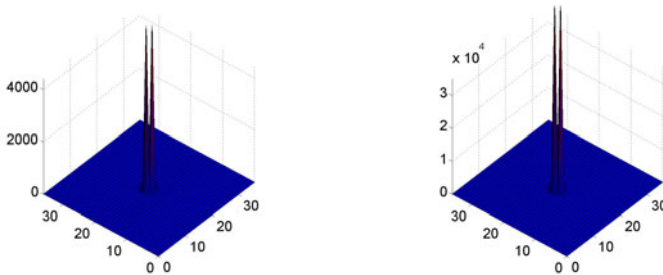


Fig. 8. Snapshot of a spatial distribution of packet queue sizes in the PSN model with $L^p_{\square}(37, QSPO, 0.040)$ setup being under DDoS attack with 6 attackers at time $k = 2^{15} = 32,768$ (left plot) and at time $k = 2^{17} = 131,072$ (right plot). The maximum packet queue size excluding the zombies’ queues is 6,158 in the left plot and 55,464 in the right plot.

with time and the number of attackers for “weak” DDoS attacks resulting in the increase of the values of the *throughput* functions, i.e. $\Theta(k) = k^{-1}N_d(k)$, as can be seen from Fig. 3 and Fig. 4. However, for “strong” DDoS attacks, i.e. with the number of attackers higher than some critical value at which the *throughput* functions attain their maximum, the values of the *throughput* functions decrease with the increase of the number of attackers, as can be seen from Fig. 4. The reason for this is that for “strong” DDoS attacks the routing loses its ability to handle efficiently the extra incoming packet traffic, the congestion spreads out outside the victim’s neighbourhood to the entire network. Thus, packets queue longer at the routers and they are not delivered in a timely manner to their destinations. This causes the degradation of the *throughput*, as can be seen from Fig. 4, the values of the *throughput* functions decrease with the increase of the number of attackers.

5 Conclusions

We have shown that the coupling of the routing with DDoS attack may have a big impact on a network performance in delivering efficiently packets to their destinations. When the network using static routing operates at *sub-critical source load* values (even when they are low) and is under a “weak” DDoS attack we have observed almost immediate degradation in the network performance as measured by the network *throughput*. The values of the *throughput* functions almost immediately decrease with each onset of a DDoS attack and they decrease with the increase of the number of attackers. However, this is not the case when instead the network uses an adaptive routing. The adaptive routing has the ability to process efficiently extra packet traffic generated by a DDoS attack when the attack is not very strong, i.e. when the number of attackers is below some critical value. For such DDoS attacks the values of *throughput* functions may even increase, i.e. when the total amount of incoming packet traffic, including the one coming from the normal incoming packet traffic and the one coming from DDoS attack, is lower than the one corresponding to the *critical source load* value. Also, we notice that when PSN model uses *ecf QS* or *ecf QSPO* the qualitative and quantitative behaviours of the *throughput* functions are very similar ones regardless of the strength of a DDoS attack. Thus, the dominant effect on the behaviour of the *throughput* functions has the dynamic cost component *QS* of the *ecf QSPO* but not the static cost component *ONE*.

Acknowledgments. The authors acknowledge the prior work of A.T. Lawniczak (A.T.L.) with A. Gerisch. A. T. Lawniczak acknowledges partial financial support from NSERC of Canada. B.N. Di Stefano acknowledges full financial support from Nuptek Systems Ltd. The authors acknowledge the use of Sharcnet computational resources and the hospitality of The Fields Institute for Research in Mathematical Sciences while writing this paper.

References

1. http://en.wikipedia.org/wiki/Denial-of-service_attack
2. http://en.wikipedia.org/wiki/ICMP_Echo_Request

3. Traynor, I.: Russia accused of unleashing cyberwar to disable Estonia. *The Guardian*, May 17 (2007), <http://www.guardian.co.uk/world/2007/may/17/topstories3.russia>
4. *The Economist*: Cyberwarfare is becoming scarier, May 24 (2007), http://www.economist.com/world/international/displaystory.cfm?story_id=E1_JNNRSVS
5. http://www.theregister.co.uk/2002/10/23/feds_investigating_largest_ever_internet/
6. http://en.wikipedia.org/wiki/Mafiaboy#cite_note-13
7. Lawniczak, A.T., Wu, H., Di Stefano, B.N.: DDoS attack detection using entropy of packet traffic in CA like data communication network model. In: Adamatzky, A., et al. (eds.) *Automata-2008 Theory and Applications of Cellular Automata*, pp. 573–584. Luni-ver Press, UK (2008)
8. Lawniczak, A.T., Wu, H., Di Stefano, B.N.: Detection & Study of DDoS Attacks Via Entropy in Data Network Models. In: *Proc. of IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA 2009)*, p. 8 (2009)
9. Lawniczak, A.T., Wu, H., Di Stefano, B.N.: Detection of Packet Traffic Anomalous Behaviour via Information Entropy. In: Fortunato, S. (ed.) *Complex Networks, CompleNet 2009. Studies in Computational Intelligence*, vol. 207, pp. 197–208. Springer, Heidelberg (2009)
10. Leon-Garcia, A., Widjaja, I.: *Communication Networks: Fundamental Concepts and Key Architectures*. The McGraw-Hill Companies, Inc., New York (2000)
11. Lawniczak, A.T., Gerisch, A., Di Stefano, B.N.: Development and Performance of Cellular Automaton Model of OSI Network Layer of Packet Switching Networks. In: *16th IEEE CCECE 2003 – CCGEI 2003*, vol. 2, pp. 1409–1412 (2003)
12. Lawniczak, A.T., Gerisch, A., Di Stefano, B.: OSI Network-layer Abstraction: Analysis of Simulation Dynamics and Performance Indicators. In: Mendes, J.F., et al. (eds.) *Proc. of AIP Conference*, New York, vol. 776, pp. 166–200 (2005)
13. Gerisch, A., Lawniczak, A.T., Di Stefano, B.: Building Blocks of a Simulation Environment of the OSI Network Layer of Packet Switching Networks. In: *16th IEEE CCECE 2003 – CCGEI 2003*, p. 4 (2003)
14. Lawniczak, A.T., Gerisch, A., Maxie, K., Di Stefano, B.: Netzwerk: Migration of a Packet Switching Network Simulation Environment from MS Windows PC to Linux PC and to HPC. In: *19th International Symposium on High Performance Computing Systems and Applications*, pp. 280–286. IEEE Press, Los Alamitos (2005)

A Cellular Automata-Based Modular Lighting System

Stefania Bandini¹, Andrea Bonomi¹, Giuseppe Vizzari¹, and Vito Acconci²

¹ Complex Systems and Artificial Intelligence (CSAI) research center
Department of Computer Science, Systems and Communication (DISCo)
University of Milan - Bicocca

Viale Sarca 336/14, 20126 Milano, Italy
{bandini, bonomi, vizzari}@disco.unimib.it

² Acconci Studio
20 Jay St., Suite #215, Brooklyn, NY 11201, USA
studio@acconci.com

Abstract. The term Ambient Intelligence refers to environments enhanced by the presence of electronic devices that are sensitive and responsive to the presence of people. The scenario described in the paper envisages an environment endowed with a set of sensors (to perceive humans or other physical entities), interacting with a set of actuators (lights) that adjust their state of illumination in an attempt to improve the overall experience of these users. The model for the interaction and action of sensors and actuators is an asynchronous Cellular Automata (CA) supporting a self-organization of the system as a response to the presence and movements of people inside it. The paper will introduce the model as well as its implementation in a specific hardware component supporting the realization of modular adaptive lighting systems.

1 Introduction

The main aim of research on Ambient Intelligence [1] is the definition of models and tools for the realization of environments endowed with a large number of electronic devices, interconnected by means of wireless communication facilities, able to perceive and react to the presence of people. These facilities can have different goals, ranging from explicitly providing electronic services to humans accessing the environment by means of computational devices (e.g. personal computers or PDAs), to simply providing some form of ambient adaptation to the users' presence (or voice, or gestures), without requiring an explicit interaction through a traditional computational device. An Ambient Intelligence system can be viewed in terms of autonomous entities, managing internal resources and interacting with surrounding ones in order to obtain the desired overall system behaviour as a result of local actions and interactions among system components. Approaches that take this perspective share a growing interest on models and mechanisms supporting forms of self-organization and management of the components (both hardware and software) of such systems.

This paper describes an asynchronous Cellular Automata (CA) based approach to the modeling and realization of a self-organizing ambient intelligence system; the latter

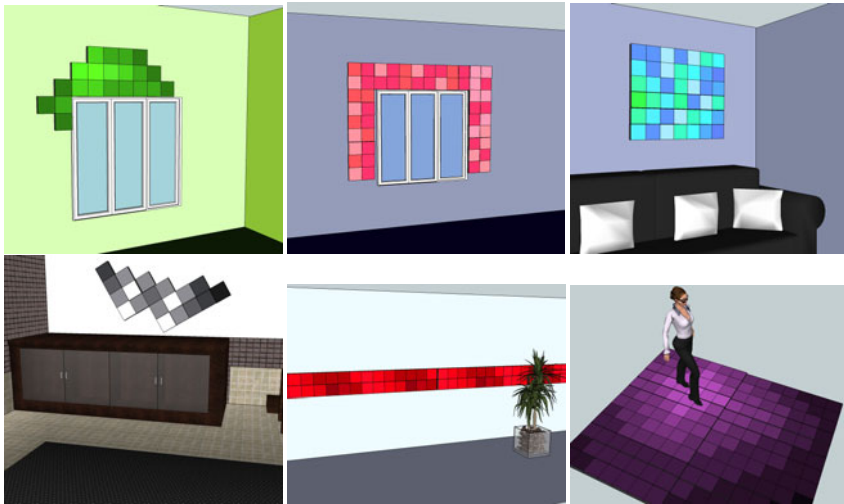


Fig. 1. Examples of different compositions of the modular Adaptive Lighting system

is viewed in terms of cells comprising sensors and actuators. The former can trigger the behaviours of the latter, both through the interaction of elements enclosed in the same cell and by means of the local interaction among adjacent cells. Since the modeled system is not necessarily characterized by the presence of a global clock synchronizing system operations we adopted an asynchronous approach to the activation of cell transition rules.

The transition rule adopted for the CA was derived by previous applications to reproduce natural phenomena such as percolation processes of pesticides in the soil, in specific percolation beds for the coffee industry and for the experimentation of elasticity properties of batches for tires [23], by modeling mechanisms of reaction and diffusion. In this specific application this rule is used to manage the interactions of cells arranged through a multilayered architecture [4], better suited to represent an artificial environment comprising a set of sensors that perceive the presence of humans (or other physical entities such as dogs, bicycles, cars), and actuators that choose their actions in an attempt to improve the overall experience of these users.

The developed model is the core component of an overall hardware-software system supporting the realization of modular adaptive lighting systems. In particular, this work represents an extension of a previous experience on the realization of a model for adaptive illumination to be adopted in the renovation of a tunnel [5]: while in the previous experience the idea was to support the realization of a large scale environment able to detect the passage of pedestrians, bicycles, cars, etc., in this case the goal was to realize modular lighting elements (i.e. simple tiles) able to decorate an environment by reacting to other sensed events or conditions like the touch of user, the passage of time, the current level of ambient light, etc. Some sample configuration of such tiles are shown in Figure 1. In particular, this work describes both the definition of a model for realizing a simple adaptation mechanism, that is similar to the one adopted in the

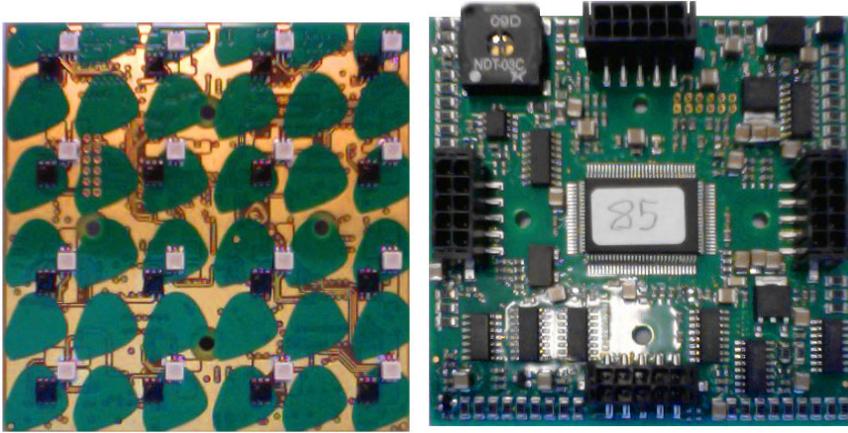


Fig. 2. The front and back sides of the module prototype. In the front picture, the white elements are the RGB leds, the black elements are the proximity sensors. In the back picture, the central element is the Fujitsu F²MC-16LX microcontroller, on the edges there are the connector, on top left the speaker.

previously mentioned experience: the mechanism is designed to realize a sort of halo effect surrounding areas stimulated by the perception of the presence of an entity (i.e. a hand or finger, in the case of a simple tile). The tiles are programmed to actually implement the model, and they are thus essentially aggregates of cells of the cellular automata.

The following Section will introduce the specific hardware platform that was adopted for the implementation of the modular lighting system. Section 3 introduces the modeling approach, setting it in the relevant literature, while section 4 describes the developed model in details, as well as its practical application in the described scenario. Conclusions and future works will end the paper.

2 Hardware Prototype

In 2009 CSAI (Complex System and Artificial Intelligence Research Center) started a collaboration with Egicon (stratEGIC innovatiON¹), an italian electronic engineering company that has been founded by people with many years of experience in electronic business. Egicon mission is to support the customer with innovative and effective solutions from the design to the production. Egicon wants to be the strategic partner for innovation.

The aim of the collaboration between CSAI and Egicon is the development of a prototype of a hardware platform suitable for a modular Adaptive Lighting. The prototypal board developed by the Egicon engineers is shown in Figure 2 and it includes the following components:

- 16 RGB leds

¹ <http://www.egicon.com>

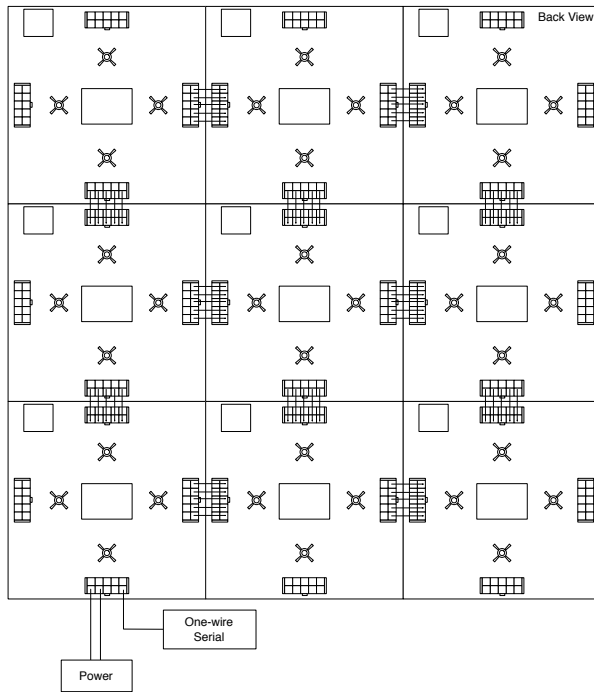


Fig. 3. Schematization of the connections between 9 modules

- 16 proximity sensors
- a speaker
- 4 communication and power connector
- a Fujitsu F²MC-16 microcontroller

The RGB led contains three leds (red, green and blue) encased in one shell. It looks like a single white led except that it has four leads - one for the common ground connection and one for each led. The current through each of the leds determines its light output (i.e. its contribution to the total output color). By controlling the current through each led it is possible to obtain different light colors.

The proximity sensors are optical proximity switch that reacts at a typical working distance of 20 mm. The sensors allow the users to interact with the Adaptive Lighting system simply touching the glass covering the lights, without pushing any buttons.

As depicted in Figure 3, the board has 4 connectors on the edges. Each connector carries both a bi-directional serial communication line and a power line, so only one of the module of an Adaptive Lighting system need to be directly connected to the power supply. Moreover each connectors provide a shared one-wire serial line for communication with an external system (e.g. a pc). This communication line is used to send command from an external controller to all the boards (e.g. for reprogramming the boards).

The board is driven by a Fujitsu F²MC-16LX microcontroller. The family of F²MC-16LX series microcontrollers serve for consumer (e.g. digital cameras, handheld

electronic product), white goods (e.g. washing machines, refrigerators), industrial (e.g. utility meters, air-conditioning systems), and automotive (e.g. body control networks, dashboards, chassis networks) applications. The MB90347 microcontroller is a 16 bit CISC running at 24MHz and has 128Kb Flash and 6Kb RAM.

3 Related Works

Cellular Automata (CA), introduced by John von Neumann as an environment for studying self-replicating systems [6], have been primarily investigated as theoretical concept and as a method for simulation and modeling [7]. They have also been used as computational framework for specific kind of applications (e.g. image processing [8], robot path planning [9]) and they have also inspired several parallel computer architectures, such as the Connection Machine [10] and the Cellular Automata Machine [11].

3.1 Asynchronous Cellular Automata

Cellular Automata have traditionally treated time as discrete and state updates as occurring synchronously and in parallel. The state of every cell of the automaton is updated together, before any of the new states influence other cells. The synchronous approach assumes the presence of a global clock to ensure all cells are updated together.

Several authors (e.g. [12][13]) have argued that asynchronous models are viable alternatives to synchronous models and suggest that asynchronous models should be preferred where there is no evidence of a global clock. Nehaniv [14] has demonstrated an asynchronous CA model that can behave as a synchronous CA, due to the addition of extra constraints on the order of updating.

Cornforth, Green, and Newth argue that asynchronous updating is widespread and ubiquitous in both natural and artificial networks [15]. They identified two classes of asynchronous behavior: Random Asynchronous (RAS), and Ordered Asynchronous (OAS) updating. Random Asynchronous includes any process in which at any given time individuals to be updated are selected at random according to some probability distribution; Ordered Asynchronous includes any process in which the updating of individual states follows a systematic pattern. An interesting study about the effects and implications of asynchronicity in CA models can be found in [16].

3.2 Dissipative Cellular Automata

The two main characteristics of the Dissipative Cellular Automata (DCA) are the asynchronous time-driven dynamics and openness: in particular, cells can be influenced by external influences coming from the environment in which they are set [17]. DCA are Asynchronous Cellular Automata: according to the asynchronous dynamics [18][19], at each time, one cell has a probability of rate λ_a to autonomously wake up and update its state.

The above characteristics are typically found in complex hardware–software systems and they can be considered as fundamental for a minimalist open agent system (or, more generally, as a minimalist open software system). The dynamic behavior of DCA is likely to provide useful insight into the behavior of real-world open agent systems and, more generally, of open distributed software systems.

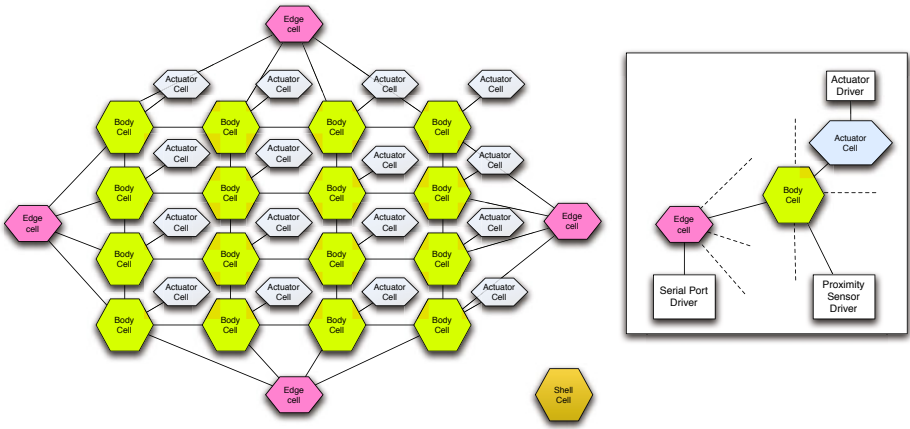


Fig. 4. On the left, the schematization of the module cells. The connections between the *Shell Cell* and the other cells are not shown for simplicity. On the right, a detail about the connection between the cells and the drivers.

3.3 Cellular Automata with Memory

Standard CA are ahistoric (memoryless): the cells have no memory of previous states, except the last one in the case the central cell is included in the neighborhood. Historic memory can be embedded in CA increasing the number of states and modifying the transaction function. Alonso-Sanz proposed to maintain the transaction rule unaltered, but make them act not only to the current state but weighted mean value of their previous states [20]. According to the author, CA with memory can be considered as a promising extension of the basic CA paradigm.

4 The Adaptive Lighting Model

The model adopted to realize the adaptive lighting behaviour combines some of the elements, mechanisms and features of the introduced CA models. In particular it realizes a Dissipative Multilayered Automata Network [21] comprising cells of different types, as depicted in Figure 4. The main cells types are:

- Body
- Actuator
- Edge
- Command Shell

In the rest of this section we describe the default behavior of such cells. Moreover it is possible to reprogram such cells in order to obtain different behaviors.

4.1 Body Cell

Each body cells is characterized by three “substance levels”. To introduce a similarity with the biological cell, each substance level represents the amount of a specific chemical substance inside the cell (e.g. CA^{++}). The three “virtual substances” are named Q_1 ,

Q_2 and Q_3 . Internally, the levels of the three substance (respectively s_1 , s_2 and s_3) are represented by an integer number between 0 and $2^{16} - 1$.

The amount of each substance is influenced by three processes:

- Stimulus Response
- Evaporation
- Diffusion

The *Stimulus Response* is the reaction of the cell to an external stimulus. The body cells are able to react to external stimuli because they are connected to the proximity sensors. Each body cell is connected to a different sensor. When a sensor is stimulated, the levels of the virtual substance of the related cell are increased in response to the stimulus. Each substance level s_i , $i \in \{1, 2, 3\}$ is incremented by a quantity defined by the parameter inc_i , an integer number between 0 and 255.

The *Evaporation* is the process of gradual disappearance of a substance, i.e. the level of the substances decreases over time. Let us define $\epsilon_i(v)$ as the function that computes the quantity of substance i to decrement from the substance level and is defined as

$$\epsilon_i(s_i) = s_i \cdot e_i^{(1)} + e_i^{(0)} \quad (1)$$

where $e_i^{(0)} \in \mathbb{R}^+$ is a constant evaporation quantity and $e_i^{(1)} \in \mathbb{R}$, $0 \leq e_i^{(1)} \leq 1$ is the evaporation rate (e.g. a value of 0.1 means a 10% evaporation rate).

The evaporation function $evp_i(s_i)$, computing the level of substance s_i from time t to $t + 1$, is thus defined as

$$evp_i(s_i) = \begin{cases} 0 & \text{if } \epsilon_i(s_i) > s_i \\ s_i - \epsilon_i(s_i) & \text{otherwise} \end{cases} \quad (2)$$

The *Diffusion* process simulates the diffusion of the substances through the cells. The body cell are disposed in a regular two-dimensional 4×4 square grid. We suppose that the cell $C_{x,y}$ is located on the grid at the position i, j , where $i \in \mathbb{N}$ and $j \in \mathbb{N}$. According to the von Neumann neighborhood [22], a cell $C_{x,y}$ has the 4 neighbors $C_{x-1,y}$, $C_{x,y+1}$, $C_{x+1,y}$, $C_{x,y-1}$. Also the cells on the border have four neighbors: one or two of neighbors are *Edge Cells*.

For simplicity, we numbered the neighbors of a cell from 1 to 4, so for the cell $C_{x,y}$, N_1 is $C_{x-1,y}$, N_2 is $C_{x,y+1}$, N_3 is $C_{x+1,y}$, and N_4 is $C_{x,y-1}$. The substance level i of the neighbor cell n is indicated with $s_{n,i}$. The mean of the substance levels sn_i of the neighbors cell is computed as:

$$sn_i = \frac{\sum_{n=1}^4 s_{n,i}}{c} \quad (3)$$

where c is the number of *connected* neighbors. A neighbor is considered connected if it is a Body Cell or if it is a Edge Cell connected to a neighborhood module.

The new value of each substance level s_1, s_2, s_3 is computed as:

$$new\ s_i = \frac{sn_i \cdot q + s_i \cdot (1 - q)}{2} \quad (4)$$

where $q \in \mathbb{R}, 0 \leq q \leq 1$ is the sensitivity coefficient (i.e. if q is equal to 0, the new state of a cell is not influenced by the neighbors values, if it is equals to 0.5 the new values is a mean among the previous value of the cell and the neighbors value, if it is equals to 1, the new value does not depend on the previous value of the cell but only from the neighbors). The computed value are rounded to integer before the assignment of the value to the substance level.

4.2 Actuator Cell

The actuators cells control the leds activities. Each actuator cell is connected to exactly one RGB led and one body cell. The actuator cell control the led according to the substance level of the Body cell. The RGB led has three independently controllable led: one red, one green and one blue. The led actuation is controlled by an intensity value between 0 and 255. The three intensity values, denoted with l_r, l_g, l_b , are computed as:

$$l_r = s_1 \cdot c_r \quad (5)$$

$$l_g = s_2 \cdot c_g \quad (6)$$

$$l_b = s_3 \cdot c_b \quad (7)$$

where s_1, s_2, s_3 are the substance amounts and c_r, c_g, c_b are the three components of the color parameter.

4.3 Edge Cell

The aim of the Edge Cells is to support communication with the other adjacent modules. Through the enclosed mirror cells, the Edge Cell makes available the relevant body cell data of the adjacent module, which are communicate by means of the serial connection. A schematization of the mirror cell is shown in Figure 5. For the body cells, the mirror cells appear identical to the other body cell, i.e. each mirror cell exposes three substances levels. The mirror cells act as a remote copies of the body cell connected on the other side of the serial connection. The activity of the edge cells can be reassumed in two tasks:

- Transmission task - During the transmission task, the values of the body cells connected to the edge cell are serialized in a message. The message is send over the serial line by the serial driver.
- Reception task - A message received through the serial driver is deserialized and the values of the body cells of the other module are copied into the mirror cell.

4.4 Prototype

The above described model was actually implemented in a C based programming environment specifically suited for the hardware platform introduced in Section 2. Two sample pictures of a combination of tiles programmed according to the above described model are shown in Figure 6. In addition to the basic model, some additional functionalities to configure the relevant parameters by means of serial communication were also

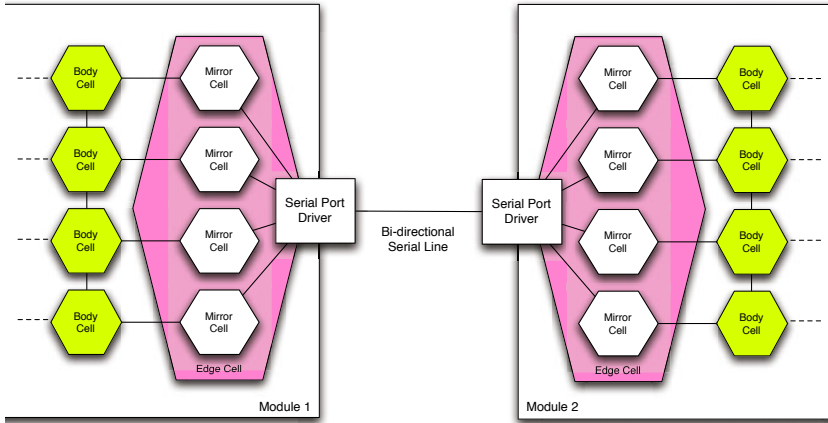


Fig. 5. Schematization of the interactions between the edge cells and the body cells

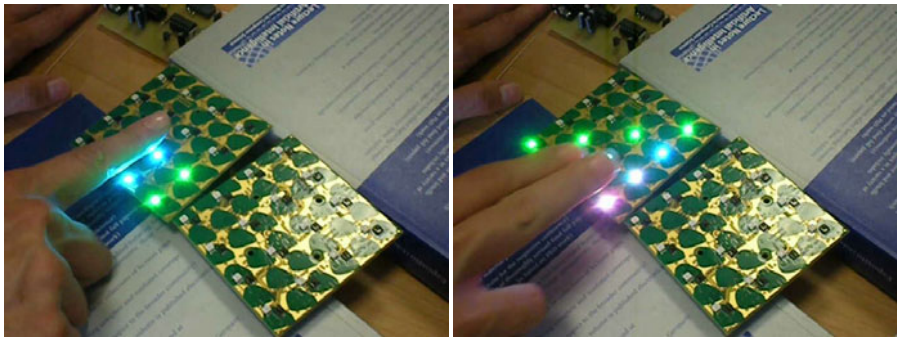


Fig. 6. Two snapshots of two tiles of the modular adaptive lighting system programmed with the described model

realized. In this way it is possible to alter the overall adaptive behaviour without re-programming the tiles (that could be not very practical and convenient, especially after their installation in a real environment), but just by interacting by means of a simple control protocol over serial communication.

5 Future Development

The paper introduced a CA based model for the realization of lighting system based on autonomous interacting modules. Each of these modules encloses a set of sensors, leds and a computational unit able to process data acquired by the sensors, manage an internal state, communicate with neighboring modules through serial lines. The overall hardware-software system employs a CA model to decide the intensity of leds according

to the dynamic state of cells in which every module is subdivided. Cells of a single module update their state according to a given transition rule that considers the previous state of the cell, the data perceived by the sensors and the state of neighboring cells, possibly situated in an adjacent module.

A prototypal hardware realization of this module was realized and described, and the modules were programmed by means of a C language based implementation of the described model. This experience led to the definition of a model and language for asynchronous automata networks. Future works include both a concrete deployment and installation of this prototype as well as further applications of the model and language that was derived by this experience.

References

1. Shadbolt, N.: Ambient Intelligence. *IEEE Intelligent Systems* 18(4), 2–3 (2003)
2. Bandini, S., Erbacci, G., Mauri, G.: Implementing Cellular Automata Based Models on Parallel Architectures: the CAPP Project. In: Malyshkin, V.E. (ed.) *PaCT 1999*. LNCS, vol. 1662, pp. 167–179. Springer, Heidelberg (1999)
3. Bandini, S., Mauri, G., Pavesi, G., Simone, C.: Parallel Simulation of Reaction-Diffusion Phenomena in Percolation Processes: a Model Based on Cellular Automata. *Future Generation Comp. Syst.* 17(6), 679–688 (2001)
4. Bandini, S., Mauri, G.: Multilayered Cellular Automata. *Theor. Comput. Sci.* 217(1), 99–113 (1999)
5. Bandini, S., Bonomi, A., Vizzari, G., Acconci, V.: An Asynchronous Cellular Automata-Based Adaptive Illumination Facility. In: Serra, R., Cucchiara, R. (eds.) *AI*IA 2009*. LNCS, vol. 5883, pp. 405–415. Springer, Heidelberg (2009)
6. von Neumann, J.: *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana (1966)
7. Weimar, J.R.: *Simulation with Cellular Automata*. Logos Verlag, Berlin (1997) ISBN 3-89722-026-1
8. Rosin, P.L.: Training Cellular Automata for Image Processing. *IEEE Transactions on Image Processing* 15(7), 2076–2087 (2006)
9. Behring, C., Bracho, M., Castro, M., Moreno, J.A.: An Algorithm for Robot Path Planning with Cellular Automata. In: Bandini, S., Worsch, T. (eds.) *ACRI*, pp. 11–19. Springer, Heidelberg (2000)
10. Hillis, W.D.: *The Connection Machine*. MIT Press, Cambridge (1985)
11. Margolus, N., Toffoli, T.: *Cellular Automata Machines*. In: *A New Environment for Modelling*. MIT Press, Cambridge (1987)
12. Paolo, E.A.D.: Searching for Rhythms in Asynchronous Random Boolean Networks. In: Bedau, M. (ed.) *Alife VII: Proceedings of the Seventh International Conference*, pp. 73–80. MIT Press, Cambridge (2000)
13. Thomas, R., Organization, E.M.B.: *Kinetic Logic: a Boolean Approach to the Analysis of Complex Regulatory Systems*. Lecture notes in Biomathematics, vol. 29. Springer, Berlin (1979)
14. Nehaniv, C.L.: Evolution in Asynchronous Cellular Automata. In: *ICAL 2003: Proceedings of the Eighth International Conference on Artificial Life*, pp. 65–73. MIT Press, Cambridge (2003)
15. Cornforth, D., Green, D.G., Newth, D.: Ordered Asynchronous Processes in Multi-Agent Systems. *Physica D: Nonlinear Phenomena* 204(1-2), 70–82 (2005)

16. Fatès, N., Morvan, M.: An Experimental Study of Robustness to Asynchronism for Elementary Cellular Automata. *Complex Systems* 16(1), 1–27 (2005)
17. Zambonelli, F., Mamei, M., Roli, A.: What Can Cellular Automata Tell Us About the Behavior of Large Multi-Agent Systems? In: Garcia, A.F., de Lucena, C.J.P., Zambonelli, F., Omicini, A., Castro, J. (eds.) SELMAS 2002. LNCS, vol. 2603, pp. 216–231. Springer, Heidelberg (2002)
18. Buvel, R.L., Ingerson, T.E.: Structure in Asynchronous Cellular Automata. *Physica D* 1, 59–68 (1984)
19. Lumer, E.D., Nicolis, G.: Synchronous Versus Asynchronous Dynamics in Spatially Distributed Systems. *Phys. D* 71(4), 440–452 (1994)
20. Alonso-Sanz, R.: The Beehive Cellular Automaton with Memory. *Journal of Cellular Automata* 1(3), 195–211 (2006)
21. Bandini, S., Bonomi, A., Vizzari, G., Acconci, V.: A CA-Based Self-Organizing Environment: a Configurable Adaptive Illumination Facility. In: Malyskin, V. (ed.) PACT 2009. LNCS, vol. 5698, pp. 153–167. Springer, Heidelberg (2009)
22. Gutowitz, H.: *Cellular Automata: Theory and Experiment*. MIT Press, Bradford Books, Cambridge (1991)

Modeling and Programming Asynchronous Automata Networks: The MOCA Approach

Stefania Bandini, Andrea Bonomi, and Giuseppe Vizzari

Complex Systems and Artificial Intelligence (CSAI) research center
Department of Computer Science, Systems and Communication (DISCo)
University of Milan - Bicocca
Viale Sarca 336/14, 20126 Milano, Italy
{bandini, bonomi, vizzari}@disco.unimib.it

Abstract. This paper introduces a model and a language for the specification and simulation of networks of automata, a generalization of Cellular Automata characterized by a possibly irregular structure, asynchronous cell transition rule activation, heterogeneity and openness to external influences. The model as well as the derived language are discussed in details, and its possible applications are briefly introduced.

1 Introduction

Cellular Automata (CA), introduced by John von Neumann as an environment for studying self-replicating systems [1], have been primarily investigated as theoretical concept and as a method for simulation and modeling [2]. CA based models have been employed for the modeling and simulation of complex systems in the most different context, from biology [3] to the social sciences [4], to traffic [5] and crowds of pedestrians [6].

In addition to these traditional fields of application, recent approaches call for the employment of CA based models for the modeling of distributed systems, for instance in the context of Ambient Intelligence [7], that are made up of similar components whose local action and interaction determines the overall system behaviour. These applications, however, generally require to relax some of the basic constraints related to CA models: from the regular lattice spatial structures to generalized graphs; from synchronous (and parallel) activation of cells' transition rules to asynchronous activation of cells' behaviours; from homogeneity of cells' states and transition rules to the possibility of defining and including heterogeneous cells in the same system. Finally, generally these systems are open to influences by external elements (e.g. sensors) and they can generate forms of actuations as a reaction to a change in the state of some specific cell.

This paper introduces a model and a language for the specification and simulation of networks of automata that can be suitably adopted to describe this kind of systems. This kind of tool can be useful to envision the effects of features like heterogeneity, openness and asynchronicity in the system by means of simulation, to effectively support the design of systems based on this approach.

2 MOCA Model

In this section we introduce the Multilayered Open Cellular Automata (MOCA) model. MOCA extends the “classic” cellular automata in several ways. The main characteristics of the models are:

- *Asynchronicity* - The cells can be updated according to several update schemes, both synchronous and asynchronous.
- *Heterogeneity* - Cells are heterogeneous, in terms of space of the states and transition rule.
- *Multilayered* - The cellular space is a hierarchical structure, deriving from the structure of the Multilayered Automata Networks [8]. A schematization of the MOCA hierarchical structure is shown in Figure 1.
- *Open* - The dynamic behavior of the automata is influenced by the external environment and influences the external environment.

Such feature are useful for designing systems composed of several distributed interacting components. MOCA can be used both to simulate the behavior of such distributed systems and to actually control the real installations. Moreover it can be used in centralized situations to control peripheral components. In the following sections we introduce the main elements of the model.

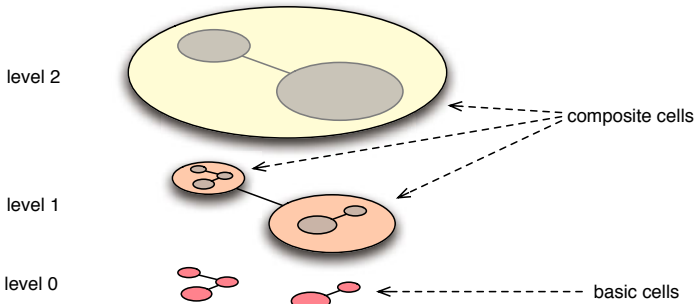


Fig. 1. Schematization of the MOCA multilayered structure

2.1 Basic Cell

As a cellular system, the fundamental element of MOCA is the *cell*. A cell can be either *basic* or *composed*.

The *basic cells* are basic building block of the MOCA. Elaborate behaviors are defined composing such basic cells. Each basic cell c , schematized in Figure 2, is characterized by:

- *Receptors* - $\mathcal{R}_c = \{\mathcal{S}_{r_{c,1}}, \mathcal{S}_{r_{c,2}}, \dots, \mathcal{S}_{r_{c,3}}\}$ a finite set of “organs” of the cell able to respond to external stimulus. Each receptor is characterized by a set of state that it can assume $\mathcal{S}_{r_{c,n}}$. The receptor can be connected to the *External state* of an other cell (in this case it assume the value of the other cells external state); in specific

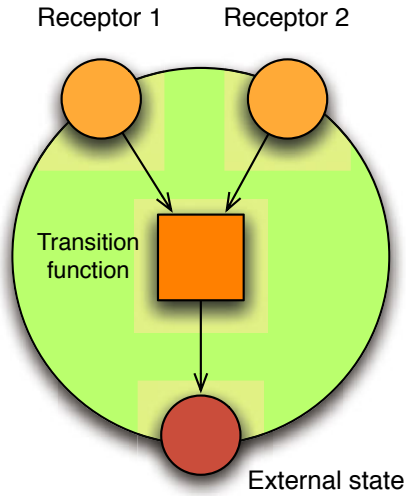


Fig. 2. Schematization of a basic cell

cases, when the basic cell is included in composite cell, the internal *receptor* could be connected to another *receptor* of the enclosing cell. A receptor not connected to any other cells assumes the *null* state.

- *External states* - $\mathcal{E}_c = \{\mathcal{S}_{e_{c,1}}, \mathcal{S}_{e_{c,2}}, \dots, \mathcal{S}_{e_{c,3}}\}$ a finite set of states “exposed” to the other cells; the external states are generally connected to the receptors of other cells; in this case, the connected receptor will assume the value of the external state of the other cell it is connected to. When the basic cell is included in a composite cell, the “internal” external state could also be connected to the “enclosing” cell’s external state.
- *Transition rule* - $f_c : \mathcal{R}_c \rightarrow \mathcal{E}_c$ determining the new value of the external states.

The receptors and externals set of states are subset of the following primitive data types:

- *int*, signed integer number
- *float*, floating point number
- *boolean* = $\{true, false\}$

For simplicity, we denoted with *number*, a signed integer number or a floating point number, and with *any* any type of data. The dimension (in terms of numbers of bits) of the numeric data types depends on the specific implementations.

MOCA is an *open system*: the dynamic behavior of the automata in is influenced by the external environment and influences the external environment. We defined the following “special” basic cells:

- *sensor cell* is a cell that can be forced by an external condition to change its state; it is provided with an external state of a given type to be connected as input to basic or composite cells.

- *actuator cell*, instead, influences the external environment according to its external state; it is provided with one or more receptors, of specific types, and it can be connected to other basic or composite cells to produce actions related to the states of the input cells.
- *open cell* is both a sensor and an actuator cell, therefore it can be provided with receptors and an external state.

A set of predefined basic cells is defined by the model. Such cells fall into a number of groups:

- Arithmetic Integer (Table 1) - These cells performs arithmetic operations on integer numbers.
- Floating Point (Table 2) - These cells performs operations on floating point numbers. These cells are available only for the MOCA Virtual Machines supporting the floating point numbers.
- Logic (Table 3) - These cells perform bitwise operations, i.e., they operate on one or two bit patterns at the level of their individual bits.
- Miscellaneous (Table 4) - Miscellaneous cells, such as *inv* cell that copies the receptor state to external state, *if*, *equal*, and *notEqual* cells.

Table 1. Arithmetic Integer Basic Cells

Name	Receptor	Transition rule	External State
add	r1:int, r2:int	$e1 = r1 + r2$	e1:int
sub	r1:int, r2:int	$e1 = r1 - r2$	e1:int
mul	r1:int, r2:int	$e1 = r1 * r2$	e1:int
div	r1:int, r2:int	$e1 = r1 / r2$	e1:int
rem	r1:int, r2:int	$e1 = r1 \text{ mod } r2$	e1:int
less	r1:int, r2:int	$e1 = r1 < r2$	e1:boolean
lessEq	r1:int, r2:int	$e1 = r1 \leq r2$	e1:boolean
great	r1:int, r2:int	$e1 = r1 > r2$	e1:boolean
greatEq	r1:int, r2:int	$e1 = r1 \geq r2$	e1:boolean

Table 2. Floating Point Basic Cells

Name	Receptor	Transition rule	External State
addF	r1:float, r2:float	$e1 = r1 + r2$	e1:float
subF	r1:float, r2:float	$e1 = r1 - r2$	e1:float
mulF	r1:float, r2:float	$e1 = r1 * r2$	e1:float
divF	r1:float, r2:float	$e1 = r1 / r2$	e1:float
lessF	r1:float, r2:float	$e1 = r1 < r2$	e1:boolean
lessEqF	r1:float, r2:float	$e1 = r1 \leq r2$	e1:boolean
greatF	r1:float, r2:float	$e1 = r1 > r2$	e1:boolean
greatEqF	r1:float, r2:float	$e1 = r1 \geq r2$	e1:boolean
iToF	r1:int	$e1 = I \rightarrow F$	e1:float
fToI	r1:float	$e1 = F \rightarrow I$	e1:int

Table 3. Logic Basic Cells

Name	Receptor	Transition rule	External State
and	r1:any, r2:any	$e1 = r1$ bitwise and $r2$	e1:any
or	r1:any, r2:any	$e1 = r1$ bitwise or $r2$	e1:any
xor	r1:any, r2:any	$e1 = r1$ bitwise xor $r2$	e1:any
not	r1:any, r2:any	$e1 = r1$ bitwise not $r2$	e1:any

Table 4. Miscellaneous Basic Cells

Name	Receptor	Transition rule	External State
inv	r1:any	$r1$	e1:boolean
equal	r1:any, r2:any	$r1 = r2$	e1:boolean
notEqual	r1:any, r2:any	$r1 \neq r2$	e1:boolean
if	r1:bool, r2:any, r3:any	$e1 = r1 ? r2 : r3$	e1:any

The set of basic cells can even be extended according to requirements of the specific applications.

2.2 Composite Cell

The composite cells are automata composed of cells, both basic and composed. Each composed cell c is characterized by:

- *Receptors* - $\mathcal{R}_c = \{\mathcal{S}_{r_{c,1}}, \mathcal{S}_{r_{c,2}}, \dots, \mathcal{S}_{r_{c,3}}\}$ are analogous to those that were defined for basic cells. Each receptor is characterized by a set of states that it can assume $\mathcal{S}_{r_{c,n}}$. Moreover, each receptor can be connected to the receptor of an internal cell.
- *External states* - $\mathcal{E}_c = \{\mathcal{S}_{e_{c,1}}, \mathcal{S}_{e_{c,2}}, \dots, \mathcal{S}_{e_{c,3}}\}$ are analogous to those that were defined for basic cells.
- *Sub-cells* - $\mathcal{C}_c = \{c_{c,1}, c_{c,2}, \dots, c_{c,n}\}$, a finite set of internal cells, not visible outside the cell but used for the internal elaboration. They can be connected to receptors of other cells or even to the external states of a comprising cell.
- *Connections* - $\mathcal{V}_c \subset \mathcal{C}_c \cup \mathcal{R}_c \times \mathcal{C}_c \cup \mathcal{E}_c$ a set of directed arcs, connecting the receptors with the sub-cells, the sub-cells among themselves, and the sub-cells to the external states.

A different update scheme can be associated to each cell. Usually, the lower layer composite cell (composed only of basic cells) adopt a synchronous scheme.

An example of composed cell is shown in Figure 3. In the example, we implement a simple component having one internal state (c_4), two receptors (r_1, r_2). The component alternatively selects one of the receptors to obtain the external state (e_1) and the new internal state by performing an *xor* of this selected input and the previous internal state.

The automaton is characterized by:

- *Receptors* - $\mathcal{R}_c = \{\mathcal{S}_{r_{c,1}}, \mathcal{S}_{r_{c,2}}\}$, where $r_1 \in \text{boolean}, r_2 \in \text{boolean}$ connected to the external state of the left and right cells.

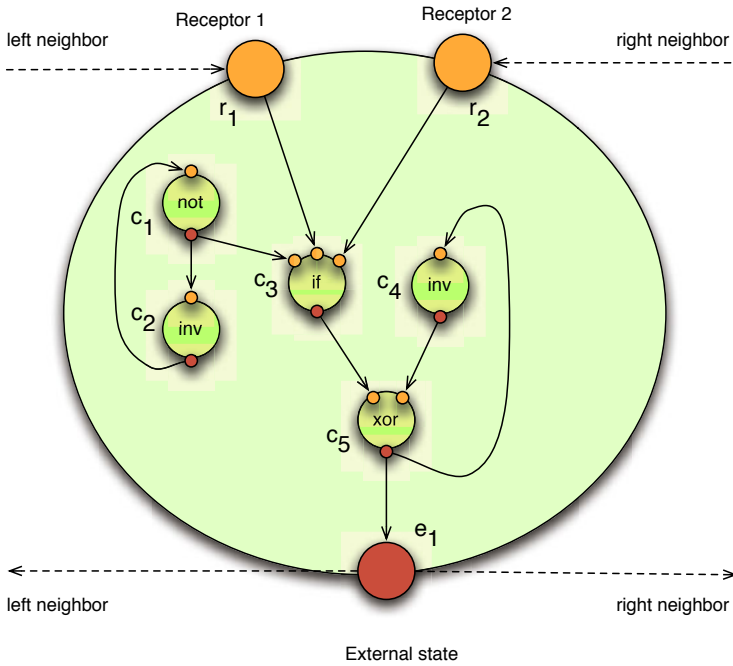


Fig. 3. Example of MOCA composed cell

- *External states* - $\mathcal{E}_c = \{\mathcal{S}_{e_{c,1}}, \mathcal{S}_{e_{c,2}}\}$, where $e_1 \in \text{boolean}$.
- *Sub-cells* - $\mathcal{C}_c = \{c_1, c_2, c_3, c_4, c_5\}$, the set of internal cells. The initial external state of all the cells is *False*.
- *Connections* - the set of directed arcs, as shown in the figure.

Two cycles of computation of this automaton are shown in Table 5

Table 5. Dynamic evolution of the automata presented in the example

t	r_1	r_2	c_1	c_2	c_3	c_4	c_5	e_1
0	T	F	F	F	F	F	F	F
1	T	F	T	F	T	F	F	F
2	T	F	T	T	T	F	T	F
3	T	F	F	T	T	T	F	T
4	T	F	F	F	F	F	F	F

2.3 Update Schemes

MOCA supports several update schemes, such synchronous, cyclic, clocked. The update scheme is determined by the parameters d_i, p_i associated to each cell i . The parameter d_i determines the delay (in terms of time step) before the first update. The parameter p_i , determines the period of the update of the cell i , i.e. how many time steps the cell i will wait in order to be updated.

The default values for the parameter are $d_i = 0$ and $p_i = 1$. This setup correspond to the *Synchronous* update scheme: all the cells are updated in parallel at each time step.

The *Cyclic* update schemes are obtained associating to each of the n cells a different value of d_i between 0 and $n - 1$, and the same period $p_i = n$.

The *Clocked* update schemes are characterized by different values of d_i and p_i . The *Equal Frequency Clocked* is characterized by different values of d_i but the same value of p_i for every cells of the automaton.

An example of such update schemes is shown in Figure 4.

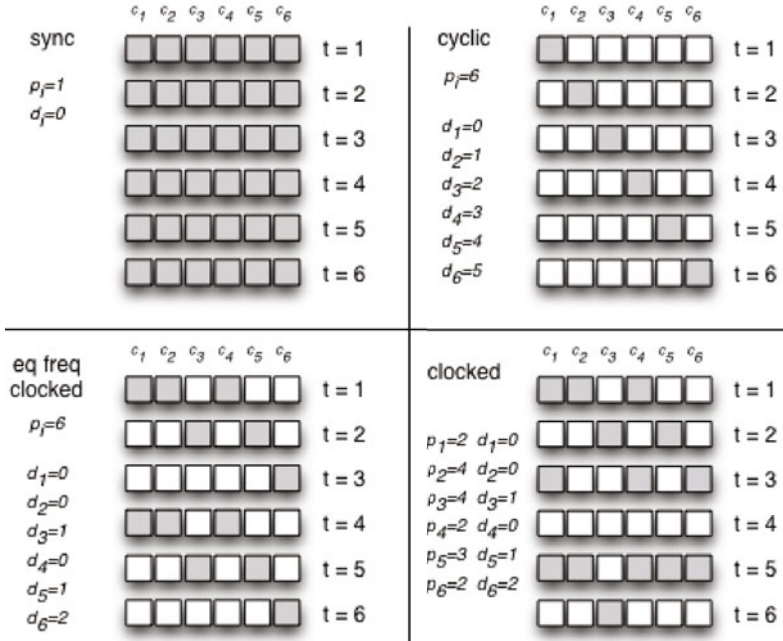


Fig. 4. An example of different update scheme obtained varying the d_i and p_i parameters

3 MOCA Programming

In this section we describe the MOCA programming language and tools. The first part of this section introduce the MOCA language, a textual cellular automata programming language. The language allows the user to create automata creating new cells and combining existing cells.

3.1 MOCA Language

The MOCA Language is a cellular automata programming language. It allows the user to create automata creating new cells and combining existing cells.

A MOCA programs consist in a set of cells descriptions. A cell description determines the *receptors*, *sub-cells*, and *external state* composing the cell and their wiring.

The order of the elements inside a cell definition is not relevance. The order of the cells definitions is relevance: all the subcell of a cell must be defined before the cell itself. An example of cell definition is the following:

```
cell rule6 {
    // Receptors and external state
    receptor boolean r1;
    receptor boolean r2;
    external boolean e1;
    // Cells
    cell not c1;
    cell inv c2;
    cell if c3;
    cell inv c4;
    cell xor c5;
    // Initial values
    c1.e1 = false;
    c2.e1 = false;
    c3.e1 = false;
    c4.e1 = false;
    c5.e1 = false;

    // Wiring
    c1.e1 -> c2.r1;
    c2.e1 -> c1.r1;
    c1.e1 -> c3.r1;
    r1 -> c3.r2;
    r2 -> c3.r3;
    c3.e1 -> c5.r1;
    c4.e1 -> c5.r2;
    c5.e1 -> c4.r1;
    c5.e1 -> e1;
}
```

The MOCA syntax is influenced by the Java syntax; it is not reported extensively for space limits. MOCA source code is *free-form* which allows arbitrary use of whitespace and ends of lines to format code. Comments may appear either between the delimiters */** and **/*, or following *//* until the end of the line.

A MOCA programs consist in a set of cells definitions. The order of the definition is not relevant. Each cell definitions begins with the *cell* keyword followed by the cells name (*identifier*) and the cell definition.

Every identifier is made from the following characters, starting with a letter:

- Letters: a–z, A–Z
- Digits: 0–9
- Underscore: _

No identifier can be the same as a MOCA keyword or pre-defined cell name. The MOCA keywords are the following: *cell*, *receptor*, *external*, *int*, *float*, *boolean*, *true*, *false*, *null*.

Cells' definitions are enclosed in braces (`{` and `}`) to limit their scope. A cell definition consists in a list of the following elements:

- *Receptor Declaration* – Declaration of a cell receptor, expressed with the following syntax:

```
receptor <receptor type> <receptor id>;
```

where the *receptor type* is one of the built-in data type and *receptor id* is an identifier. The keywords *int*, *float*, and *boolean* specify built-in data types.

- *External State Declaration* – Declaration of a cell external state, expressed with the following syntax:

```
external <ext. state type> <ext. state id>;
```

where the *ext. state type* is one of the built-in data type and *ext. state identifier* is an identifier.

- *Sub-Cell Declaration* – Declaration of a sub-cell, expressed with the following syntax:

```
cell <cell type> <cell id>;
```

where the *cell type* is an identifier of a pre-defined or a user-defined cell type and *cell id* is an identifier for the sub-cell. The sub-cells scope is limited to the cell definition.

- *Initial Value Assignment* – Assignment of the initial external state of a sub-cell, expressed with the following syntax:

```
<sub-cell id>.<sub-cell ext. state id> =  
    <literal>;
```

where *sub-cell id* is an identifier of a sub-cell, *sub-cell ext. state id* is an identifier of an external state of the sub-cell, and *literal* is the value to be assigned to the external state of the cell. A *literal* is the source code representation of a value of a built-in data type. The literals are the *numeric literals*, floating-point numbers expressed according to the Java syntax, *boolean literal*, *true* and *false*, and the *null literal*, represented by the keyword *null*.

- *Initial Delay Assignment* – Assignment of the initial delay before the first update of a sub-cell, expressed with the following syntax:

```
<sub-cell id>.delay = <number>;
```

where *sub-cell id* is an identifier of a sub-cell and *number* is the initial delay in terms of time step.

- *Period Assignment* – Assignment of the update period of a sub-cell, expressed with the following syntax:

```
<sub-cell id>.period = <number>;
```

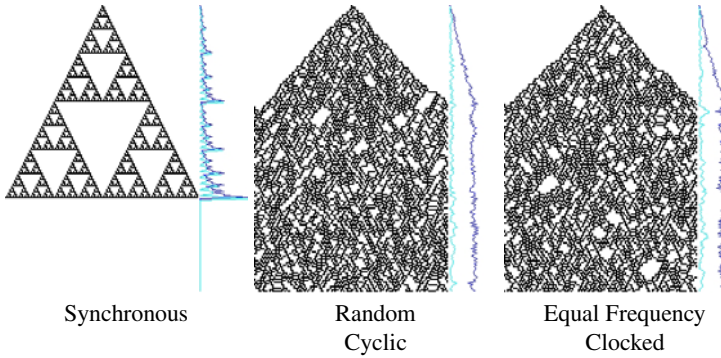


Fig. 5. Time space diagrams a 1D CA based on the sample MOCA composed cell

where *sub-cell id* is an identifier of a sub-cell and *number* is the length of the period expressed in time step.

- *Wiring* – Definition of an arc between two cells, or a receptor and a cell or a cell an an external state. Direct arc between a receptor and an external state are not allowed.

```
<source> -> <target>;
```

where *source* can be a receptor identifier or a sub-cell receptor identifier and the *target* can be an external state identifier of a sub-cell external state identifier. The sub-cell receptor and external state identifier are expressed with the following syntax:

```
<sub-cell id>.<receptor/external state id>
```

4 Current and Future Applications

An interpreter of the introduced language has been developed and it has been used to simulate the effects of asynchronicity in systems based on MOCA cells. Figure 5 shows a comparison of the time-space diagrams of different one-dimensional systems composed of the sample MOCA composite cells introduced in Section 2.2 with the Synchronous update scheme, the cell produces a dynamic evolution similar to the *Sierpinski Triangle* fractal. This typical shape is not present with any of the other update schemes. Moreover if the automaton has periodic boundaries conditions and the number of cells is a power of two, starting from an initial configuration, the evolution of the synchronous automata eventually reaches the fixed point. The automata with the other update schemes does not have this behavior.

In addition to the interpreter, a compiler and a virtual machine for supporting the programming of specific autonomous hardware modules whose behaviour is specified according to the model have been developed. Two sample applications of this model and language are the control system for an adaptive illumination facility [7] and a modular lighting system [10].

5 Conclusions and Future Works

This paper introduced a model and a language for the specification and simulation of networks of automata, a generalization of Cellular Automata characterized by a possibly irregular structure, asynchronous cell transition rule activation, heterogeneity and openness to external influences. An interpreter of the language has been developed and it has been used to simulate the effects of asynchronicity in systems based on MOCA cells.

Future works are directed, on one hand, towards the application of this model and language to the realization of self-organizing modular illumination systems, extending the work described in [7]; on the other hand this model and language represent the empirical counterpart (in the vein of [9]) of a parallel and coordinate line of work characterized by an analytic nature aimed at evaluating the possibility of effectively assuring specific relevant global properties for asynchronous CA systems.

References

1. von Neumann, J.: Theory of Self-Reproducing Automata. University of Illinois Press, US (1966)
2. Weimar, J.R.: Simulation with Cellular Automata. Logos Verlag, Berlin (1997) ISBN 3-89722-026-1
3. Christley, S., Zhu, X., Newman, S.A., Alber, M.S.: Multiscale Agent-Based Simulation for Chondrogenic Pattern Formation *n vitro*. *Cybernetics and Systems* 38(7), 707–727 (2007)
4. Hegselmann, R., Flache, A.: Understanding Complex Social Dynamics: a Plea for Cellular Automata based modelling. *J. Artificial Societies and Social Simulation* 1(3) (1998)
5. Nagel, K.: Cellular Automata Models for Transportation Applications. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) ACRI 2002. LNCS, vol. 2493, pp. 20–31. Springer, Heidelberg (2002)
6. Burstedde, C., Kirchner, A., Klauck, K., Schadschneider, A., Zittartz, J.: Cellular Automaton Approach to Pedestrian Dynamics - Applications. In: Pedestrian and Evacuation Dynamics, pp. 87–98. Springer, Heidelberg (2001)
7. Bandini, S., Bonomi, A., Vizzari, G., Acconci, V., DeGraaf, N., Podborseck, J., Clar, J.: A CA-Based Approach to Self-Organized Adaptive Environments: The Case of an Illumination Facility. In: Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2008, pp. 1–6 (October 2008)
8. Bandini, S., Mauri, G.: Multilayered Cellular Automata. *Theor. Comput. Sci.* 217(1), 99–113 (1999)
9. Fatès, N., Morvan, M.: An Experimental Study of Robustness to Asynchronism for Elementary cellular automata. *Complex Systems* 16(1), 1–27 (2005)
10. Bandini, S., Bonomi, A., Vizzari, G., Acconci, V.: A Cellular Automata-based Modular Lighting System. In: Bandini, S., et al. (eds.) ACRI 2010 proceedings, pp. 337–347 (2010)

Efficient Circuit Construction in Brownian Cellular Automata Based on a New Building-Block for Delay-Insensitive Circuits

Jia Lee^{1,2} and Ferdinand Peper²

¹ College of Computer Science, Chongqing University, Chongqing, China

² Nano ICT Group, National Institute of Information and Communications Technology, Kobe, Japan

Abstract. A Brownian cellular automaton (BCA) (Lee & Peper, 2008) is an asynchronous cellular automaton, where configurations representing signals propagate randomly in the cellular space, resembling particles under Brownian motion. Depending on merely three kinds of local transition rules, this BCA model can be used to construct all primitives in an universal set of delay-insensitive circuit elements, such that all well-known logic circuits can be realized in the cellular space. Though Brownian-like movements of signals enable spontaneous searching for solutions through the state space of computation, their diffusive behavior may induce substantial overhead in the operation of the circuits. In this paper, we propose a novel kind of primitive element that can be employed as a new building-block for the delay-insensitive circuits. Moreover, construction of this new element in the BCA can utilize the spontaneous fluctuations of signals more effectively, while at the same time restricts the Brownian behavior into possibly smaller configurations as compared to the construction of the previous elements, thus will improve the efficiency of the realized logic circuits in the BCA.

1 Introduction

A *cellular automaton* (CA) is a discrete dynamical system consisting of a huge number of identical finite-state automata (cells) that are locally connected in a uniform way. As a generalized model of CAs, asynchronous cellular automata (ACA) allow cells to be updated independently at random times. Computation in ACAs usually requires some special mechanism to control the unpredictable update nature of cells, which in turn usually causes the increase in the complexity of ACAs as compared to their synchronous counterpart. Nevertheless, inclusion of fluctuation into ACAs promises models with less complexity, e.g., the *Brownian cellular automaton* (BCA [4,5]) which allows local configurations-like signals-to run back and forth randomly on wires, as if they were subject to Brownian motion.

In spite of the randomness of the fluctuations of signals, it actually forms a powerful resource that can be employed to backtrack circuits out of deadlocks and to equip the circuits with arbitration ability [8]. As a result, the BCA uses

merely three kinds of transition rules to update the states of cells, which is far less than achieved thus far in literature for computationally universal ACA models (e.g. [3]). Furthermore, each primitive element from an universal set of delay-insensitive (DI) circuit elements can be constructed respectively by local configurations within the cellular space. A DI-circuit is a kind of asynchronous circuits whose correct operation is robust to arbitrary delays on wires or elements [1]. Thus, all logic circuits can be embedded straightforward into the BCA by placing local configurations of each primitive at appropriate positions, followed by connecting them with wires, according to their design schemes by DI-circuit elements [4].

The above constructions of logic circuits confine all necessary Brownian motions to local configurations representing primitive elements, and hence, no longer needs backward propagation of signals on a wire between two elements. This allows the placement of Ratchets on the wire to speed up the signals. Though Brownian motions of signals allow spontaneous searching for solutions through the computational state space, they usually tend to slow down the operation of the circuits (BCA), because of the overhead caused by the diffusion-like propagation of signals. Thus, how to restrict the Brownian behavior into smaller local configurations representing primitive elements becomes crucial to further improve the efficiency of logic circuits constructed in the BCA.

In this paper, we propose a new primitive element for DI-circuits, which will take the place of an old building-block in the above universal set. We show that the construction of our novel element in the BCA can utilize the spontaneous fluctuations of signals more effectively, while at the same time restrict the Brownian behavior into smaller spaces as compared to the construction of the replaced element. Thus, logic circuits such as the AND, OR gates and the one-bit Adder circuit can be constructed in the BCA based on the new element, and operate more efficiently than their previous constructions.

This paper is organized as follows: Section 2 gives an overview of DI-circuits and introduces our new build-block. Decomposing the new building-block into Brownian circuit elements is described in Section 3, in accordance with which the construction of logic circuits in the BCA model is shown in Section 4. This paper finishes with the conclusions given in Section 5.

2 Delay-Insensitive Circuits and a New Building-Block

A delay-insensitive (DI) circuit is a kind of asynchronous circuits, whose correct operation is robust to arbitrary delays involved in lines or elements. Communications between the circuit and the outside world are done via exchanging tokens(signals) through the input and output lines. A DI-circuit is called *conservative* if it conserves the number of tokens between inputs and outputs.

Like conventional synchronously-timed circuits, all conservative DI-circuits can be constructed from a fixed set of primitive elements. Figure 1 gives some examples of primitive elements, in which the Merge, Conservative 2x2-Join and CSequencer that can be used to realize any arbitrary conservative DI-circuit, i.e., they form a

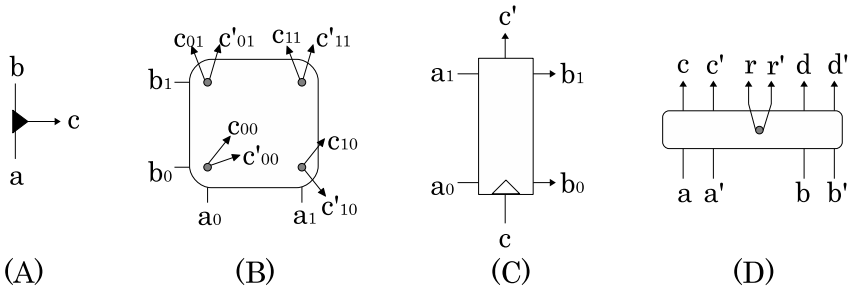


Fig. 1. Examples of conservative DI-circuit elements. (A) **Merge**: A signal arriving on input line a or b is transferred to output line c . (B) **Conservative 2x2-Join**: A pair of signals with one arriving on input line a_i and another one arriving on b_j ($i, j \in \{0, 1\}$), is assimilated and results in one signals on each of the output lines c_{ij} and c'_{ij} . (C) **CSequencer**(Conservative Sequencer): An input signal on line a_0 (resp. b_0) together with an input signal on line c_0 are assimilated, resulting in an output signal on line b_0 (resp. b_1) and on line c_1 . If there are input signals on both a_0 and b_0 at the same time as well as an input signal on c_0 , then only one of the signals on a_0 and b_0 (possibly chosen arbitrarily) is assimilated together with the signal on c_0 . The remaining input signal will be processed at a later time, when a new signal is available on line c_0 . (D) **DRJoin** (Dual Resettable Join): This is a new building-block for conservative DI-circuits. A pair of signals with one arriving on input line a (or b) and another one on input line a' (resp. b'), is processed and results in two signals each on output lines c (resp. d) and c' (resp. d'), respectively. A signal arriving on input line a or a' together with another signal arriving on line b or b' , are assimilated and give rise to two signals each of output line r and r' , respectively.

universal set [6]. A conventional design scheme for DI-circuits uses the CSequencer to serialize parallel arrivals of input signals, as well as use the Merge as a fan-in element. In addition, a Conservative 2x2-Join is employed as a fundamental logic operator and plays the key role in accomplishing more complicated logic operations, and hence, it serves as a building-block for DI-circuits.

For example, Fig. 2(A) shows the realization of a Boolean XOR gate by the Conservative 2x2-Join. Here we employ the *dual-rail encoding* to represent the binary input and output bits of the XOR gate, i.e., using two lines to encode a bit, this encoding scheme represents the value 0 by a signal on one line and the value 1 by a signal on the other line. Moreover, A one bit Half-Adder is a well-known logical circuit that performs an addition operation over two one-bit binary numbers, and outputs a sum of the two inputs and a carry. Figure 2(B) illustrates that a Half-Adder can be constructed straightforward using the Conservative 2x2-Join.

In this paper, we propose a new primitive element for conservative DI-circuits. Called DRJoin (see Fig. 1(D)), this primitive element can be used to construct logic circuits, such as the XOR gate and the Half-Adder circuit (see Fig. 3), with the constructions being as efficient as those based on the Conservative 2x2-Join. Furthermore, as implied by Fig. 3(C), any arbitrary conservative Di-circuit

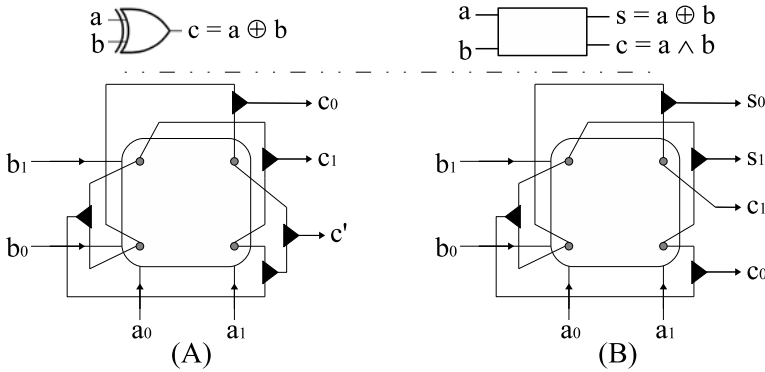


Fig. 2. (A) A Boolean XOR gate and the realization of a dual-rail XOR gate by Conservative 2x2-Join and Merge. The output line c' is used to output the surplus signal in response to each pair of input signals, because the number of signals is conserved throughout the whole construction. (B) A Half-Adder circuit and the realization of a dual-rail Half-Adder based on Conservative 2x2-Join and Merge.

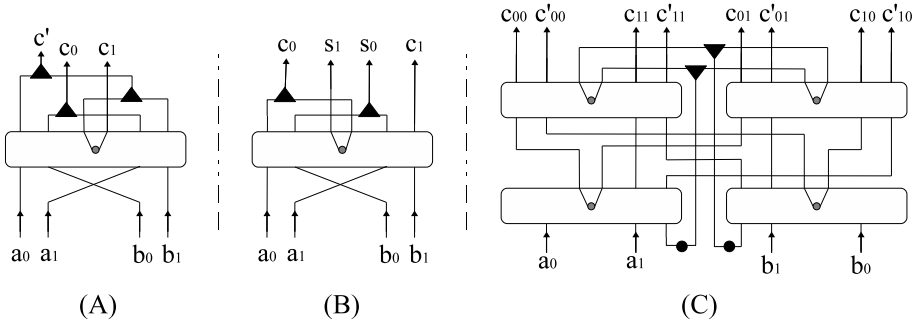


Fig. 3. (A) Construction of the dual-rail XOR gate in Fig. 2(A) by DRJoin and Merge elements. (B) Construction of the dual-rail Half-Adder in Fig. 2(B) by DRJoin and Merge elements. (C) Realization of the Conservative 2x2-Join by DRJoin and Merge, in which the two black blobs represent two signals being assigned initially on the lines.

can be realized by the Merge, DRJoin, and CSequencer elements, i.e., {Merge, DRJoin, CSequence} is universal.

3 Decomposing New Building-Block into Brownian Circuit Elements

A *Brownian circuit* [8,5] is a special type of conservative DI-circuit, which allows the movements of tokens(signals) on lines fluctuate randomly between going forward and backward. The possible reversal movements of tokens enable the circuit to backtrack from the deadlock states. Deadlocks may take place in token-based asynchronous systems, like the Petri net, and usually requires special

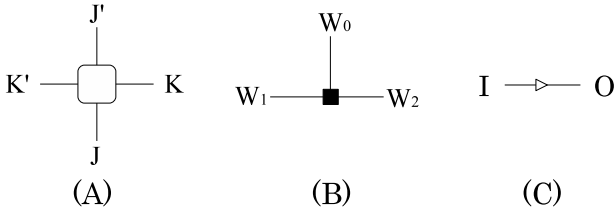


Fig. 4. Primitive elements of Brownian circuits. (i) **CJoin** (Conservative Join): Two signals with one arriving on line J (or K) and another one on line J' (resp. K') are processed and give rise to two signals each on one of the lines K and K' (resp. J and J'), respectively. (ii) **Hub**: A signal arriving on line W_i will be transferred to one of the other lines W_j , where $i, j \in \{0, 1, 2\}$ and $i \neq j$. (iii) **Ratchet**: This element works as a diode such that once a signal pass through it, the signal can not pass the element any more in the backward direction.

functionality in the systems to resolve them. Random fluctuations of tokens, however, can provide this functionality as part of their nature, thus allowing for simpler primitive elements and circuit constructions.

As a result, three kinds of Brownian circuit elements each of which has much simpler functionality than those primitives given in Section 2, are shown to form a universal set from which any arbitrary conservative DI-circuits can be constructed [5,4] (see Fig. 4). In addition, the constructions of the Merge, Conservative 2x2-Join, and CSequencer elements are shown in Fig. 5.

The bold dotted lines in Fig. 5 are used to denote those areas without the placement of Ratchets, so that signals arriving on them will run randomly fluctuating between the forward and backward directions. Though such Brownian motion-like movements of signals are essential to realize the functionalities of logic circuits, as shown in Fig. 5, they may cause substantial overhead into the processing times of the circuits. Since propagations of signals resemble the diffusion processes of

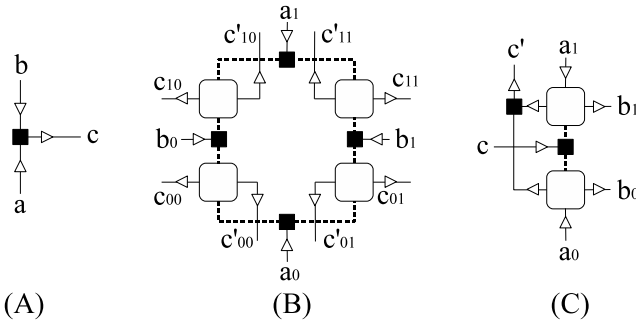


Fig. 5. Realizations of (A) Merge, (B) Conservative 2x2-Join, and (C) CSequencer elements by CJoin, Hub and Ratchet. For simplicity, we exploit the bold dotted lines to denote those lines on which the placement of Ratchet is not allowed.

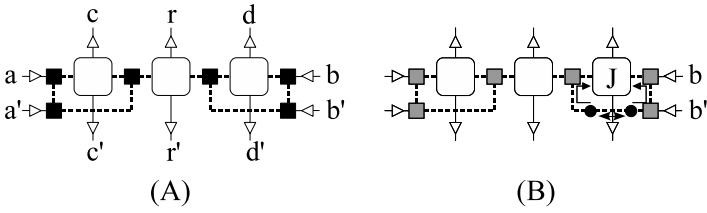


Fig. 6. (A) Decomposing DRJoin into CJoin, Hub, and Ratchet elements. (B) When two signals with one received from the input lines b and another one received from b' enter into the circuit, the possible repulsive force between them will push the two signals apart so that they may be processed by the CJoin J more easily.

particles, reducing the spaces where signals fluctuate will significantly improve the efficiency of the operations of the constructed circuits.

Figure 6(A) shows the realization of a DRJoin by the CJoin, Hub and Ratchet elements. The construction in Fig. 6(A) may enable more efficient physical implementation, for example, as implementation in cellular automata, in the sense that sizes of the areas prohibiting the placement of Ratchet (lengths of the bold dotted lines) can be shortened as compared to the construction in Fig. 5(B). Moreover, another potential advantage of the construction in Fig. 6(A) is that it allows two signals run simultaneously on the same line, as illustrated in Fig. 6(B), such that if the two signals repel each other in a similar way as the well-known repulsion occurring between two electrons, then the mutual repulsive force between signals may actually suppress their random behavior and accelerate their processing by the circuit.

4 Embedding New Building-Block-Based Circuits into Brownian Cellular Automata

A *Brownian cellular automaton* (BCA) is a 2-dimensional cellular automaton in which each cell assumes a state from the state set $\{0, 1, 2\}$, denoted by white, gray, and black, respectively. Each cell does state transitions based upon the state of itself, along with the states of the four adjacent cells in non-diagonal directions (the von Neumann neighborhood). The local transition function is described by transition rules each of which is given in the form as shown in Fig. 7(A), such that a rule not only changes the state of the central cell, but also the four neighbors of the cell, like a block cellular automaton. Furthermore, the local transition function is rotational symmetrical, i.e., for any rule in the function, its rotational equivalence (see Fig. 7(B)) is included in the local function, too.

Cells are updated in accordance with three kinds of transition rules: R_1, R_2, R_3 , as listed in Fig. 8, together with their rotational equivalences. Moreover, transitions of the cells take place asynchronously as follows: At each time step, one cell is selected randomly from the cellular space. If the state of the selected cell and the states of its neighboring cells match the lefthand side of a transition rule, then the rule is applied to update the states of the cell as well as its neighbors.

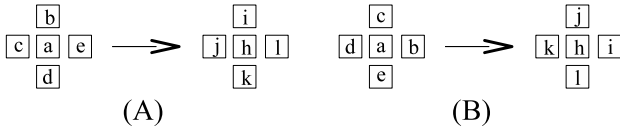


Fig. 7. (A) A transition rule where $a, b, c, d, h, i, j, k \in \{0, 1, 2\}$, and (B) its rotational equivalence, in which both the lefthand and righthand sides of the rule on the left were rotated 90 degree in the clockwise direction

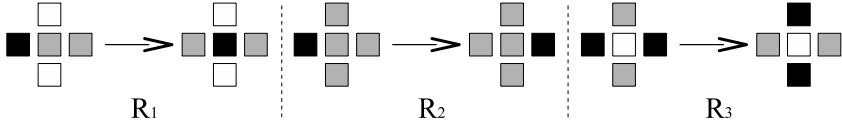


Fig. 8. List of transition rules, with their rotational equivalences left out

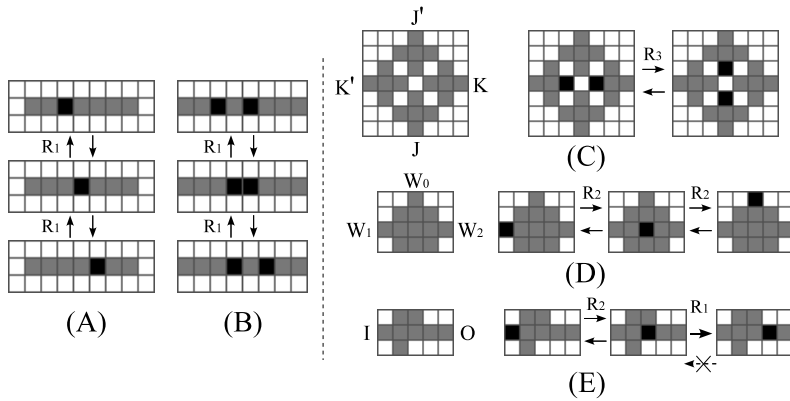


Fig. 9. (A) Random propagation of a signal on a straight line. (B) Collision of two signals running on the same line. Local configurations representing (C) CJoin, (D) Hub, and (E) Ratchet, along with the processing of signals arriving on their input/output lines. For simplicity, transition rules that are used to update cells within a configuration are denoted along with the arrows.

A line in the BCA is represented by a continuous sequence of cells in state 1, and a signal is denoted by a cell in state 2 on a line, as illustrated in Fig. 9(A). Due to the transition rule R_1 in Fig. 8 (or its rotational equivalences), the movement of a signal will fluctuate randomly between going forward and backward directions. Moreover, multiple signals can appear on the same line and propagate independently. In this case, when two signals collide with each other on a line, the only possible update of cells will pull one cell away from the other cell (see Fig. 9(B)), which seems like a kind of repulsion force occurring between the signals.

In addition, local configurations that are used to represent the CJoin, Hub, and Ratchet elements are shown in Fig. 9(C)–(E), respectively. Thus, according

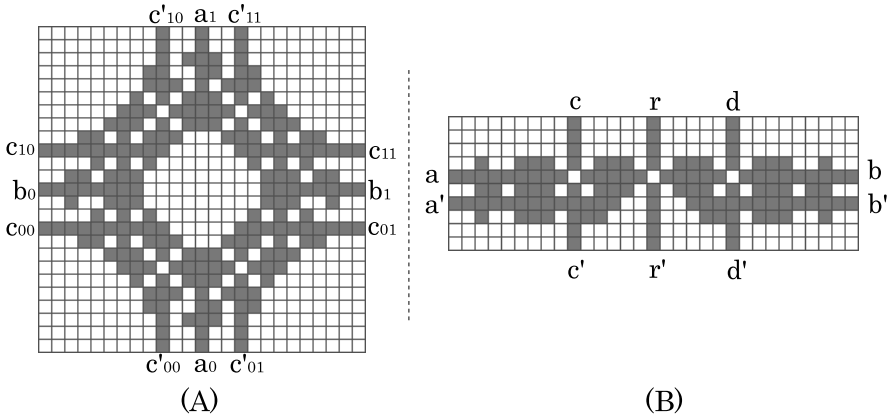


Fig. 10. Local configurations representing (A) Conservative 2x2-Join and (B) DRJoin elements, respectively

to the design schemes given in Figs. 5(B) and 6 respectively, it is able to construct the Conservative 2x2-Join and DRJoin elements in the BCA, as shown in Fig. 10.

Comparing the two local configurations in Fig. 10, obviously the construction of DRJoin restricts indispensable random movements of input signals into smaller regions than the construction of Conservative 2x2-Join. Furthermore, suppose two signals received from input line a (or b') and a' (resp. b') respectively enter into the configuration of DRJoin. Because a signal can not leap across another signal during their running on the same line (see Fig. 9(B)), such repulsion-like interference between them may effectively speed up the processing of the two signals by the DRJoin element. As a result, logical circuits like the XOR gate and Half-Adder circuit constructed by the DRJoin can operate more efficiently than their constructions by the Conservative 2x2-Join element.

5 Conclusions

Asynchronous cellular automata are a promising architecture for future nanocomputers [2,7], because they may allow bottom-up fabrication based on molecular self-assembly. The inclusion of fluctuation in cellular automata actually reduced the complexity of the BCA model substantially, as compared to non-Brownian cellular automaton models. The key to success is due to the Brownian circuit elements, by which any arbitrary DI-circuit can be constructed. Thus, in accordance with the decomposing schemes of Di-circuits into Brownian circuit elements, it is able to implement any logical circuit in the BCA.

In order to further improve the computational efficiency of the BCA, we proposed a new building-block for Di-circuits in this paper. This new building-block enables its construction in the BCA operate more effectively, in the sense that it confine the Brownian movements of signals to possibly smaller spaces as compared to the construction of the previous building-block. In addition, mutual

repulsion between signals can be exploited in the construction to accelerate their arrivals at desired destinations respectively, and hence, improving the efficiency of signal processing.

Finally, the single electron technology (SET) offers potential for implementation of Brownian circuits [9]. In this case, witness the reduced complexity of the construction by Brownian circuit elements, as well as the possibility of active exploitation of the repulsive force between electrons, our new building-block for Di-circuits may improve the feasibility of SET technology for implementing Brownian circuits.

Acknowledgements

The authors are grateful to the anonymous referees for their useful comments and suggestions. This research work was partially supported by the Fundamental Research Funds for the Central University (No. CDJRC10180008), China.

References

1. Hauck, S.: Asynchronous design methodologies: an overview. *Proc. IEEE* 83(1), 69–93 (1995)
2. Heinrich, A.J., Lutz, C.P., Gupta, J.A., Eigler, D.M.: Molecule cascades. *Science* 298, 1381–1387 (2002)
3. Lee, J., Adachi, S., Peper, F., Mashiko, S.: Delay-insensitive computation in asynchronous cellular automata. *Journal of Computer and System Sciences* 70, 201–220 (2005)
4. Lee, J., Peper, F.: On Brownian cellular automata. In: *Theory and Applications of Cellular Automata*, p. 278. Luniver Press (2008)
5. Lee, J., Peper, F., et al.: Brownian circuits—part II (in preparation)
6. Patra, P., Fussell, D.S.: Conservative delay-insensitive circuits. In: *Workshop on Physics and Computation*, pp.248–259 (1996)
7. Peper, F., Lee, J., Isokawa, T.: Cellular nanocomputers: a focused review. *Int. J. of Nanotechnology and Molecular Computation* 1, 33–49 (2009)
8. Peper, F., Lee, J., et al.: Brownian circuits—Part I (in preparation)
9. Saffiruddin, S., Cotofana, S.D., Peper, F., Lee, J.: Building Blocks for Fluctuation Based Calculation in Single Electron Tunneling Technology. In: *Proc. 8th IEEE Conf. Nanotechnology (Nano 2008)*, TX, pp. 358–361 (2008)

A Cellular Automaton Controlled Shading for a Building Facade

Machi Zawidzki

Ritsumeikan University
Kusatsu, Shiga, Japan
zawidzki@gmail.com

Abstract. A practical implementation of cellular automata (CA) in the field of architecture is presented, where a CA drives a modular shading system of a building facade. Application of a 2-color, 1- dimension, range-2 (2C-1D-R2) CA on a square grid for a regular type of a building facade and the prototype of a CA controlled shading device is presented. The problem of average grayness of a pattern is presented. The solutions for problems of linear gradual change of average grayness as a function of sequence of initial conditions and the sequence of initial conditions which cause desired opacity change of the shading array are presented. The fabrication of the CA shading prototype consists of: design of the logic circuits, fabrication of the units, followed by design of the LCD panel and the acrylic casings.

1 Introduction

Interesting qualities of cellular automata (CA) astonish for decades, however their practical (physical) applications besides as pretty pictures are still rather sparse. Already in 1940s John von Neumann designed the Universal Constructor—a self-replicating machine in a CA environment without the use of a computer [1]. However, in fact practically all the CA activity is confined to the virtual world of computers. On the other hand for quite a while architects have been dreaming of architecture that can change its appearance. One of the most recognized examples is Jean Nouvel’s Arab World Institute in Paris, where 30,000 light-sensitive diaphragms were installed on the south facade [2]. The motivation for such a change ranges from rational adaptation to environmental variation according to a time of day, season, temperature etc. to pure aesthetics and ordinary commercialism. This paper presents a modular shading system that changes the average opacity of the building facade and takes visual advantage of the emerging behavior of a CA as shown in Figure 1.

The most important reasons for applying cellular automata (CA) in this case are:

- Emergent behavior manifested by generated complex geometrical pattern
- Modularity with every cell having the same structure
- Low cost

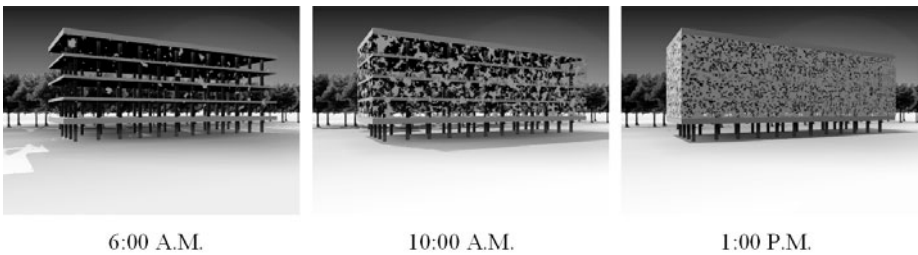


Fig. 1. Visualization showing organic behavior of a building facade, where opacity is controlled in relation to the daylight conditions. The facade evolves to maintain a constant level of light indoors at changing luminosity level of outdoor light.

- Robustness of the system which will operate despite local damage to the array
- Flexibility allowing CA to be applied on any topologically regular grid- not only rectangular.

Although the presented array is two dimensional, the CA used are one-dimensional. The convention for presenting a one-dimensional CA is to show the history of generation changes, where each row corresponds to a step in the history. Every row becomes the initial condition for the next row and so forth. Use of a two-dimensional CA (2D CA) may seem more intuitive because the domain of 2D CA is greater than 1D, so the chances of finding an amazing CA are greater, the inter-cell wiring seems to be easier and so on. Nevertheless, there are major concerns involved with the application of 2D CA. The most difficult problem is that, although their behavior is often truly amazing, it is difficult to control the states of cells. The 2D CA continuously updates all cells until it reaches equilibrium which almost always leads to an uninteresting, mostly uniform state, that is all the cells in this case remain black or white with some artifacts left over (small islands of the opposite color) and often locally strobing cells (switching the state at every step forever). Usually, the final state of the whole array is difficult or impossible to predict due to the computational irreducibility, or not useful for the shading device due to the strobing effect. Controlling the state of 2D arrays is very difficult or perhaps impossible. This problem could be solved by freezing the array at a certain step and not allowing it to evolve further, but at present it seems to be a difficult technical problem. With the adopted common convention of displaying 1D CA, this problem does not occur, because every row displays the state at a certain step, so set once- maintains the state. The second major problem with 2D CA is the setting of the initial conditions. How to set the initial input to the given cells of a 2D array? It seems possible to use only cells on the edges of the array- as it is done in the 1D case, but further investigation and experimentation is required. In the presented case of a 1D CA, a row of cells receives input from the row above and becomes the initial condition for the row below and so forth. This process continues in a cascade and propagates down the whole array of cells as shown in Figure 2.

At first the investigation for the proper CA is shown followed by the fabrication process of the prototype.

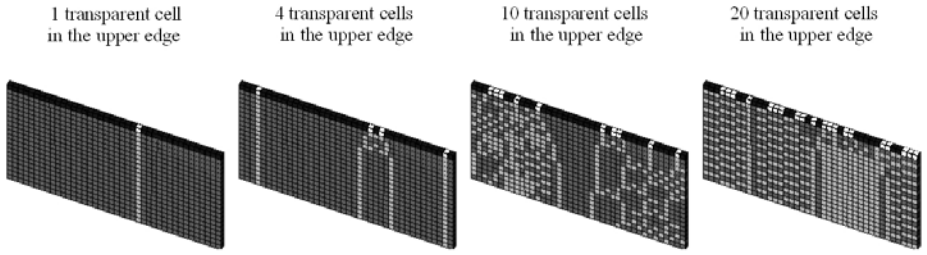


Fig. 2. The overall grayness of the array is controlled by the top row. Four different initial conditions are shown: 1, 4, 10 and 20 transparent cells in the top row.

2 Search for the Proper CA

A proper CA for application on a building facade is defined as: rendering wide range of grays, controllable (monotonic grayness function) and visually appealing.

2.1 The Grayness Function

The grayness function is the relationship between the number of black cells in the initial condition to the number of black cells in the whole array. Figure 3 shows an example of a class 4 CA demonstrating interesting patterns, but unsuitable for shading purposes- the grayness function is not monotonic.

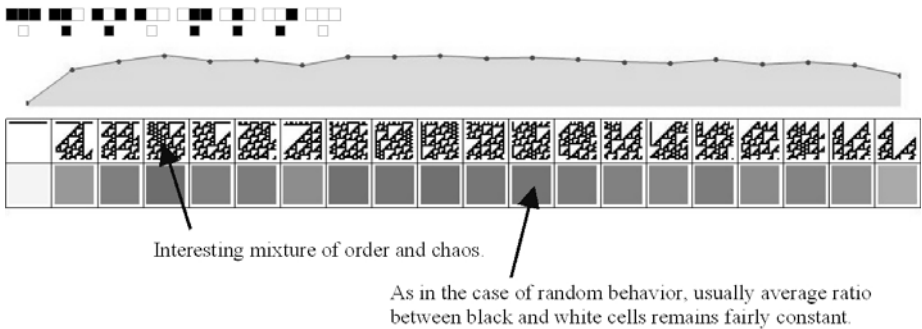


Fig. 3. A Class 4 EA (Rule 110): The pattern generated is interesting visually, however it seems impossible to control the average grayness of the array

2.2 Rule 3818817080,2,2

Since among elementary cellular automata (EA) there are no rules that generate interesting patterns and at the same time render the proper grayness function, the investigation moved towards more complicated ones, hoping that in greater number of possible CA there will be some that meet both of the given criteria. This could be done by increasing the number of possible states of a cell (colors),

increasing the dimension or as it is done in this project- by widening the size of the neighborhood. The search was based on rule symmetry [3], which means that for inverted initial conditions, the generated patterns will be exactly inverted. After finding a number of symmetric rules, test for grayness function was applied. Final selection was done arbitrarily according to the visual attractiveness of the pattern as shown in Figure 4.

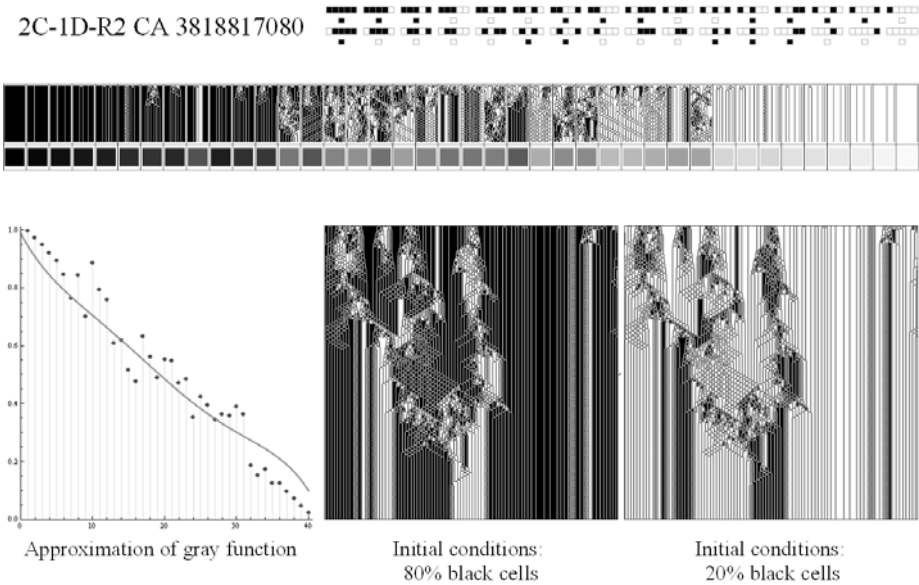


Fig. 4. Rule 3818817080,2,2. On the top: set of rules; below: incremental change from 100% to 5% black cells in the initial conditions showing both patterns and an average gray below; bottom from the left: the grayness curve and two symmetric sample patterns

2.3 The Sequence of Initial Conditions

Setting the initial conditions is as equally important as finding the appropriate rule and must meet two constraints: the k^{th} initial condition has exactly one black cell less than $(k - 1)^{th}$ and preserves all the rest of the black cells. Such a way ensures that the changes of the shading array will not appear excessively chaotic or disturbing and the transition from one state to another can be understood and rationally interpreted by the observer. Experiments showed that generally, the changing sequence should be fairly scattered, since it usually produces more interesting patterns than a consecutive way.

3 The Prototype

Since it is one of the first engineering projects involving a physical device using the concept of a CA, the prototype intends for demonstrative purposes to be

not only a shading device, but to explain the idea of a CA. Instead of realizing the selected CA, and since it was decided to make the actual hardware, it was rational to build a universal circuit capable of demonstrating all four classes of CA behavior. A universal elementary CA unit circuit capable of emulating any of 256 EA was designed. In order to show other possible applications of the idea the low-tech approach was applied.

3.1 The Logic for the CA Module

The logic for an electrical circuit- an analogue of a cellular automaton cell was designed as shown in Figure 5.

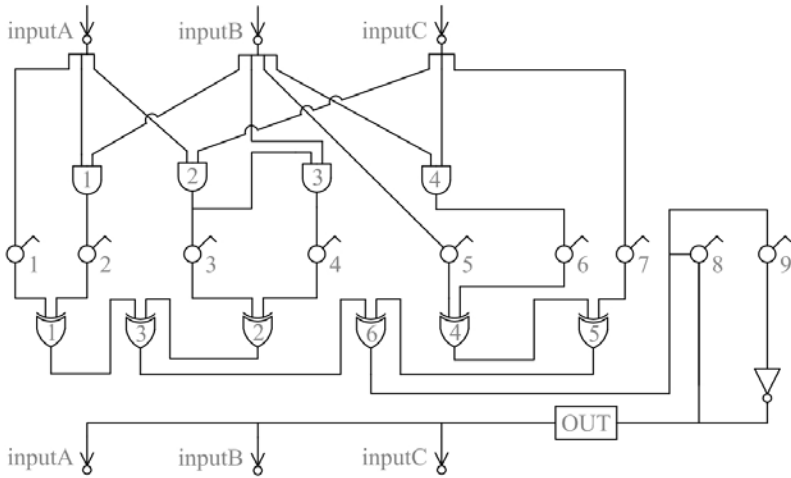


Fig. 5. The smallest logic to emulate any of 256 elementary automata

3.2 An Electrical Circuit for a CA Cell

An electrical circuit based on this logic scheme utilizes four dual AND (using all the gates of a Quad 2-Input AND CMOS), six dual XOR using all one Quad XOR CMOS and only half of the gates of the second Quad XOR CMOS, one Inverter gate using only one of four gates of a Hex INV CMOS and nine switches (integrated 10-DIP switch with nine out of ten switches used). Each circuit was equipped with 8 SIP resistor (for eight switches) + 4 resistors (ground for all four CMOS) to prevent from floating in the circuit, and a LED lighting set with LED, transistor and two resistors. The electric diagram of the circuit is shown in Figure 6.

3.3 A 3x8 CA Array

To document all 256 EA by generating 256 unique patterns it is sufficient to use just 8 output cells of the second step, but for demonstration purposes, a bigger

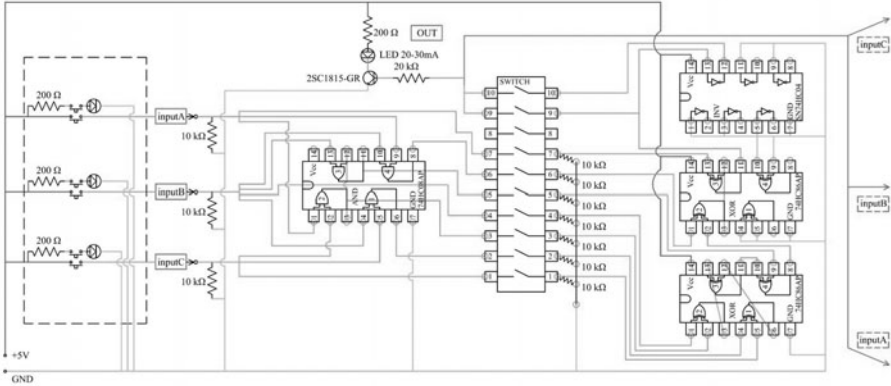


Fig. 6. The electrical diagram of the CA unit. The dotted rectangle on the left contains an additional LED which was used for the test unit only.

array was built. The prototype consists of four rows, where the first row will become the manually set initial condition, and the next three upper rows will demonstrate three consecutive CA generation steps. Using periodic boundary conditions, twenty four modules and eight input switches were connected into a hardware CA array. For the ease of manipulating the input (CA initial conditions), the orientation of the CA grid was reversed, that is the input cells are on the bottom, the first step is located above and so forth as shown in Figure 7. A special acrylic casing was designed and manufactured.

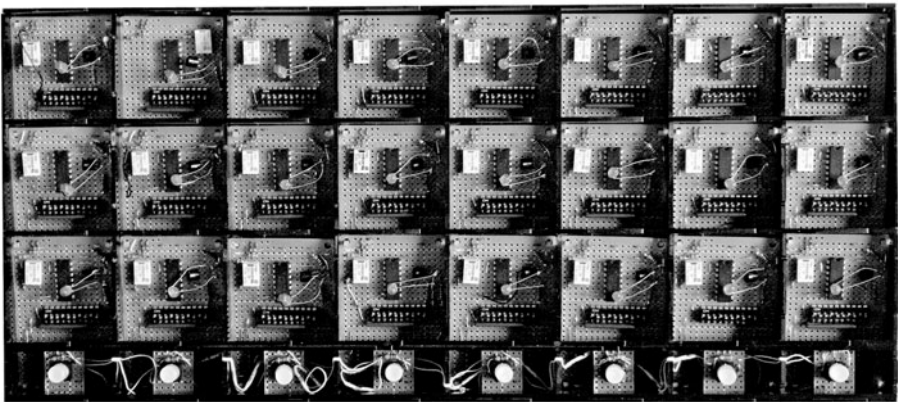


Fig. 7. A photograph of the assembled array of twenty four CA cells (+ eight initial input cells on the bottom)

3.4 Playing with CA

The twenty four cell CA device can demonstrate the concept of elementary cellular automaton by a "hands-on" experience. After removing the front cover-manual switching the settings of every cell allows students to understand how complex behavior (of the whole array) emerges from multiple interferences of simple "decisions" taken by individual cells. The device is shown with LCD panel in Figure 8.

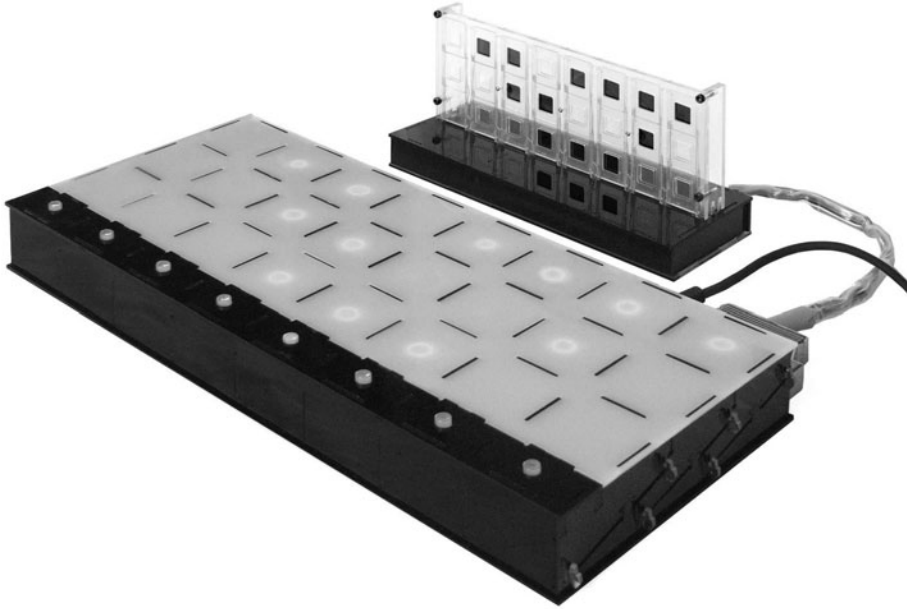


Fig. 8. A photograph of the CA shading prototype. The LCD shading panel is connected to the CA device. Rule 30 EA at one cell input.

3.5 LCD Shading Panel

For demonstrating the shading action, LCD technology was used. The light transmission through a liquid crystal cell can be varied from 0 to 100% by the bias. The position of the two polarizers determines whether the cell is normally-black (0% transmission at 0V) or normally-white (100% transmission at 0V). The single layer of film has approximately 38% transmittance for unpolarized light. Two sheets parallel (transparent state) have an average transmission of 27% and crossed (opaque state)- 0.04%. At 550 nm, the center of the visible spectrum, the crossed transmission is less than 0.01%. When 5V are applied on the unit, the cell turns black (opaque), otherwise it is transparent. For proper operation and longevity, LCDs require an alternate voltage application, thus a special inverter circuit was designed and built. Every CA cell was slightly modified and equipped

with a relay switch. A special acrylic casing was designed and manufactured for the LCD panel. The complete system is shown in Figure 8.

4 Conclusions

- By implementation of cellular automata, it is possible to control the average opacity of a shading array and create very interesting patterns at the same time.
- The project demonstrates a physical device based on a cellular automaton with possible practical application.
- The prototype was made inexpensively in the university laboratory [4], using very basic tools and skills.
- Since the scale of the considered application to a building facade is rather large- the cost of a unit is an important issue. The circuit for a single CA (even range-2) would be much simpler than the universal unit presented in this paper, therefore mass production of an optimized integrated circuit will result in a significantly lower price for each unit.

Acknowledgments

- I am grateful to Japanese Ministry of Education, Culture, Sports, Science and Technology for the support of my research, to Professor Kazuyoshi TATEYAMA and Professor Tomonori HANASAKI for the support and guidance, to Stephen Wolfram for inspiration, to Ed Pegg Jr and Todd Rowland for help and sharing of ideas.
- I would like also to thank Sir Yoshiro TORII for his invaluable help, patience and friendship and to the students of the Professor Fujieda Laboratory who helped me with soldering the CA units: Tatsuya MASADA, Yasunori INOUE, Yuji KAWAI, Yasuhiro SUGIMOTO, Takuya DOKI and Tadayuki KANEMURA.

References

1. Von Neumann, J., Burks, A.: Theory of Self-Reproducing Automata. University of Illinois Press, Urbana (1966)
2. The official website of the Arab World Institute, <http://www.imarabe.org>
3. Zawidzki, M.: An implementation of cellular automata for dynamic shading of a building facade. *Complex-Systems* 18(3) (2009)
4. Zawidzki, M.: The prototyping of a shading device controlled by a cellular automaton. *Complex-Systems* (2010) (in print)

FPGA Design of a Cellular Automaton Model for Railway Traffic Flow with GPS Module

Anastasios Tsiftsis, Georgios Ch. Sirakoulis, and John Lygouras

Laboratory of Electronics, Department of Electrical and Computer Engineering,
Democritus University of Thrace, 67100 Xanthi, Greece
{atsiftsi,gsirak,ilygour}@ee.duth.gr

Abstract. In this paper, an electronic system able to reproduce the complex dynamic behaviors of the train movement is presented. In particular, a Cellular Automaton (CA) model inspired by Li et al. corresponding model was developed in order to provide efficient control of the railway traffic flow. The proposed model was implemented on a (Field Programmable Gate Array) FPGA to take full advantage of the inherent parallelism of CAs. The FPGA design which results from the automatically produced synthesizable VHDL code of the CA model is considered as basic component of a portable, low total cost electronic system. The later also includes a high performance Global Positioning System (GPS) wireless communication module for the monitoring of train activity in the under study railway. The aforementioned module in conjunction with the proposed fully automatically programmable FPGA device minimizes the design burden offering the chance of real-time train control operation based on the presented CA model.

Keywords: Cellular Automata, FPGA design, Train operation control, GPS.

1 Introduction

Railway transport is a major form of passenger and freight transport in many countries all over the world. Railway transportation is capable of high levels of passenger and cargo utilization and energy efficiency, but is often less flexible and more capital-intensive than highway transportation is, when lower traffic levels are considered. On the other hand, railways are certified safe land transportation systems when compared to other forms of transportation [1]. Trains can travel at very high speed, but they are heavy, unable to deviate from the track and require a great distance to stop. Possible accidents include derailment (jumping the track), a head-on collision with another train and collision with an automobile or other vehicle at level crossings. In order to maximize their overall performance as a trustworthy, safe and accurate transport media, the train operation control and, in particular, construction and mainly the coordination of train schedules and plans for existing railway networks gathers more attention worldwide. Consequently the need for appropriate computational tools and

models able to optimize the usage of the existing railway networks is perpetual and imperative.

During the previous years, sufficient number of works has been published regarding the comprehension, modeling and simulation of the train control operation system [2,3]. Additionally, some mathematical models have been proposed for the optimization and control of train movement under different control system conditions [4]. However, taking into account the numerous complex constraint conditions associated with the under study mathematical model, the solution of the proposed model often demands extensively high usage of computational resources. In general, the simulation models for train control system can be classified into three types, i.e. the basic model, the time-based model, and the event-based model [5]. The basic model is suitable for simulating the train control system when the length of track conductor loop is small. The time-based approach is easy to design and build simulation models, but it needs a high computational demand. The advantage of the event-based model is that it can save computational effort, but it reduces the accuracy.

In the view of foregoing and taking into account that rail transit system, including traffic environment, railway system, control system, and individual element aggregation of train stream, can be easily considered as a dynamic complex system, some models and methods for simulating the railway traffic based on Cellular Automata (CAs) have been proposed. This alternative arrives from the fact that CAs are very effective in simulating systems and solving scientific problems, because they can capture the essential features of systems where global behavior arises from the collective effect of simple components which interact locally. As a result, CAs have been used in the modelling and simulation of railways since they can dynamically simulate the departure time, route choice and the delay propagation of trains as well as simulate different types of railway traffic by simply modifying the basic rules of each proposed CA model. Li et al. presented a simplified one-dimensional CA model to the analysis of train tracking and railway traffic flow for the first time [6,7], which originated from the well-known NaSch model of road traffic [8]. The proposed model able to research the relationship between micro-laws and macro-measures, is suitable for investigating the railway traffic, and then simulated the train flow near a railway station and analyzed the characteristics of railway train tracking. Recently, almost the same authors proposed a CA to simulate the tracking operation of trains in Beijing subway line 2 [5] and a station model based on CAs which was composed of the two tracks, i.e. the main track and the siding track [9]. Spyropoulou [10] proposed a model for the simulation of traffic at signalized intersections which was also originates from the NaSch model [8]. Finally, Tomoeda et al. [11] presented a real-time Tokyo Metro Railway Network simulation tool named "KUTTY".

In this paper, a CA model for the railway traffic flow, able to simulate the complex dynamic behaviors of the train movement, is proposed. The proposed model was inspired by the Li et al. model [5,6,7] providing new features to overcome some of the previous model limitations; for example, the original rules have to be modified and extended in order to successfully simulate more realistic

situations of train traffic flow near the station. On the other hand, the presented CA model is characterized by as much as low complexity as possible so that the computational recourses are kept low while its computation speed is kept high. Furthermore, one of the most pronounced features of the introduced model is its ability of stand alone train control operation without any need of central operator [7] which in some means degrades the CA local attitudes.

Moreover, because of the inherent parallelism of CAs, the proposed model is hardware implemented with the help of Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) synthesizable code in order to speed up the application of CAs to the study of trains' movement. In particular, for design automation reasons, a compilation function automatically exploits the CA parameters values previously determined by the user and results in synthesizable VHDL code that describes the CA model. It should be mentioned that CAs are one of the computational structures best suited for hardware realization. The CA architecture offers a number of advantages and beneficial features such as simplicity, regularity, ease of mask generation, silicon-area utilization, and locality of interconnections [12]. As a result, no need for high silicon overhead is required for the implementation of the aforementioned circuit. In this paper, the design processing of the finally produced VHDL code, i.e. analysis, elaboration and simulation, has been checked out with the help of the Quartus II, v. 7.2 design software of the ALTERA Corporation. Test benches were automatically constructed by our system, for the simulation needs of the VHDL code, and the Simulator of Quartus was used to simulate the operation of the dedicated processor described by the VHDL code obtained. Consequently, the implementation of the resulting VHDL code results in a single Field Programmable Gate Array (FPGA) processor, which is considered as basic component of an electronic system able to provide real time information concerning the accurate speed of each of the moving or stopped trains of the examined railway network and able to handle the train control system. More specifically, taking into account GPS (Global Position System) tracking with the help of a miniaturized sensor, the resulted FPGA processor is fed with real data about the trains speed and feedbacks with a support decision system providing on time information regarding the possible optimal control of the railway network. The proposed electronic system is also equipped with wireless transceiver-transmitter for communication reasons as well as with proper detectors corresponding to possible obstacles found in the train route. As a result, the proposed FPGA design could serve as the basis of a support decision system for monitoring train movement in real-time, providing valuable near optimum control services.

In the length of this paper, details about the proposed CA model and the resulting simulation results are found in Section 2. The corresponding FPGA architecture of the proposed model and its automation design procedure are discussed in Section 3, while the proposed GPS sensor design is described in full details in Section 4. Finally, the conclusions are drawn in Section 5.

2 The Proposed CA Train Control Model

CAs models have formed the theory for the development of several transportation models to simulate various types of elements such as vehicles, pedestrians or even railway traffic. In the NaSch traffic model [8], the network is represented by assigning to each link a number of cells that have a specific length (that number would result from the division of the length of the link by the space value of the cells). The dynamics of the model are described by four simple rules which are: 1) Find number of empty sites ahead ($=gap$) at time t . If $u > gap$ (too fast), then slow down to $u = gap$. 2) Else if $u < gap$ (enough headway) and $u < umax$, then accelerate by one: $u = u + 1$. 3) If after the above steps the velocity is larger than zero ($u > 0$), then, with probability p , reduce u by one. 4) Each particle moves u sites ahead: $x=x+u$.

Taking into account the NaSch as well the resulting Li et al. train model, in the proposed model 1-d CA model, the lane consists of a single lane which is divided into L cells of equal size numbered by $i = 1, 2, \dots, L$, and the time is discrete. Each site can be either empty or occupied by a train with integer speed $v_n = 0, 1, \dots, v_{max}$ or by a station or by a station with a train stopped at it. It should be also mentioned that the probability found in Rule 3 is not taken under consideration and its value equals to zero. For each n train the following parameters are taken into account: i) D_x the distance between the n train and the train immediately ahead, i.e. the distance headway, ii) x_s the distance of the n train from the next station, iii) L the number of the CA cells corresponding to the railway lane (in the proposed CA model only one single lane is considered), iv) x_c the distance needed for the train to slow down (break) in order to stop in front of the station, v) v_n the velocity of the n train, vi) d_n the minimum instantaneous distance, vii) s_m the safety distance, ix) a the acceleration rate of each train, x) b the deceleration rate of each train and xi) t_d the station dwell time, after which the train leaves the station.

If a CA cell is occupied by a train n then the states of the cells ahead are checked for the presence of other trains. In case there is a leading $(n+1)^{th}$ train, and the distance between the two trains, D_x is larger than distance d_n , the n^{th} train is accelerated by a . On the other hand, if distance D_x is smaller than distance d_n , the n^{th} train is decelerated by b . Of course, in case the two distances equal, the velocity of the n^{th} train remains the same. When a station is located before n^{th} train and distance x_s equals to x_c , the train will decelerate in order to reach the station with proper zero velocity. Finally, when every cell ahead is free, the n^{th} train is accelerated to reach its maximum velocity. In the proposed model, the minimum instantaneous distance d_n is automatically adjusted and depends on the train velocity at each time step. For the calculation of this distance the worst scenario is taken into account for safety reasons. In particular, the train is decelerated continuously by b and the resulting distance d_b is summed up with the distance covered by the train in the specific velocity v , d_v :

$$d_n = d_v + (d_v - d_b) + (d_v - 2d_b) + (d_v - 3d_b) + (d_v - 4d_b) + \dots \quad (1)$$

In this equation the distance d_v is summed up due to the fact that in case the distance is minimum, the velocity remains the same. Another case that should be taken under consideration is when the train is accelerated and is not able to stop. As a result, the minimum distance d_n should be recalculated for this case and another check is desired for the train in order to accelerate or decelerate. Following the aforementioned considerations, distance x_c can be adjusted accordingly. Furthermore, as mentioned before when a station is occupied by a train this should remain at the station for t_d , which also depends on the number of passengers at the station. It should be mentioned that in the proposed model the time interval during which the train remains at station after halting, is independent of the size of waiting crowd of passengers, assuming that there is sufficiently large number of broad doors in the trains [13]. When this period finishes a check is needed regarding the distance from the next occupied cell, in other words, if it is more or less than, $d_{nplus} = (d_a + d_b)$, a new safety distance for the train located on station. On the other hand, when this check results in bigger distance, then the train will exit the station with the minimum acceleration a . Consequently, the train will be able to slow down before any possible occupied cell. In different case, thus the distance is less than d_{nplus} the train remains in the station for more time. When the next occupied cell is a station, the minimum distance is decreased to d_a in order to exit one station and enter immediately the other. Obviously, if no cell is occupied the train will leave the station with acceleration a .

The proposed model works as described by the following pseudocode:

Case 1, the train n is behind the train $n - 1$

Acceleration of the n^{th} train:

```

If  $Dx_n > d_n$ 
    then  $v_n = \min(v_n + a, v_{max})$ 
ifelse  $Dx_n < d_n$ 
    then  $v_n = \max(v_n - b, 0)$ 
else  $v_n = v_n$ 
    
```

Slowing down of the n^{th} train:

```

 $v_n = \min(v_n, Dx_n)$ 
    
```

Movement of the n^{th} train:

```

 $x_n = x_n + v_n$ 
    
```

Case 2, the train n is behind a station.

In case the station is occupied by a train the rules are the same as applied in Case 1.

In case the station is empty

Acceleration of the n^{th} train:

```

If  $x_s > x_c$ 
    then  $v_n = \min(v_n + a, v_{max})$ 
ifelse  $x_s < x_c$ 
    
```

```

then  $v_n = \max(v_n - b, 0)$ 
else  $v_n = v_n$ 

```

Slowing down of the n^{th} train:

```

 $v_n = \min(v_n, Dx_n)$ 

```

Movement of the n^{th} train:

```

 $x_n = x_n + v_n$ 

```

Case 3, the train n is behind a station.

(1) The n^{th} train is behind another train:

```

If  $d_x > d_n$  plus
  then  $v_n = a$ 
else  $v_n = v_n$ 

```

(2) The n^{th} train is behind another station:

```

If  $d_x > a$ 
  then  $v_n = a$ 
else  $v_n = v_n$ 

```

Some simulation results are depicted in space-time diagrams of Fig. 1 where x -axis indicates the direction of train movement, and the y -axis corresponds to time evolution. More specifically, a CA grid with 300 cells was considered, while the time steps equal to $T = 300$. Several stations as well as trains are considered at different cells and different distances and consequently stop and go waves

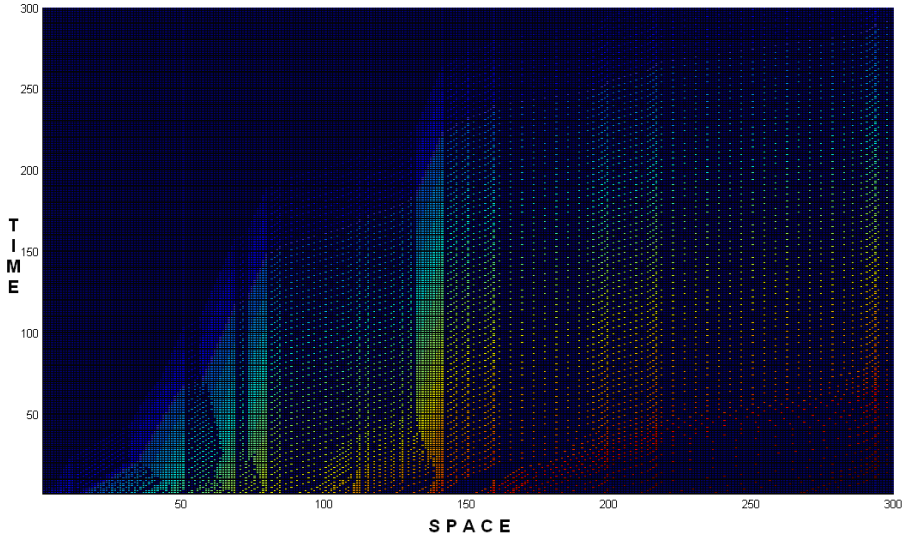


Fig. 1. Local space-time diagrams of traffic flow of $L = 300$ cells for $v_{max} = 10$, $a = 2$, $b = 1$, $t_d = 1$

have been noticed. Of course, the later is related to several parameters of the CA model, beyond the train and station density, however more details can be found in [5,10]. The results can be found in qualitative agreement with the ones found in relative references [5,6,7,8,9,10].

The most engaging difference between the presented method and the previous ones of Li et al. is the absence of any need of a central or/and area computer in order the CA model to control the train movement. As shown in the next sections an electronic system with the help of GPS can be used in order to implement the presented model which can be under conditions fully auto-controlled. Furthermore, as mentioned before, the minimum instantaneous distance d_n as well as distance x_c are automatically adjusted at each time step based on previous equation. Some further changes have been implemented taking under consideration special cases such as boundary conditions that could help the overall performance of the system. In the proposed system, in order to “help” the train located at the CA boundaries to exit the CA grid with proper velocity, empty cells are virtually added. Additionally, extra checks about zero velocity $v_n = 0$ and minimum velocity $v_n = 1$ have been added for different cases. For example in Case I, in acceleration phase, the first check is enhanced as follows: *If $Dx_n > d_n$ or $v_n = 0$* ; while an extra check for the minimization of the time needed for a train to reach an obstacle cell with the help of the appropriate distance and velocity conditions has been also added. Finally, in case of presence of tunnels along the railway tracks, where the velocity calculation would be difficult due to the limited receiving of the GPS device, the trains are moving in the tunnels with invariant velocity, i.e. the velocity they have reached at the beginning of each tunnel.

3 FPGA Implementation

To take advantage of the natural parallelism of CAs models, synchronous very large scale integrated (VLSI) circuits should be used for their implementation. Furthermore, the hardware implementation of these models could be achieved after the manual translation of their parts into a synthesizable subset of VHDL. The top level of the proposed CA cell digital design consists on basic functional VHDL modules in correspondence to the simulated train network as depicted in Fig. 2. More specifically, each block, namely **Check**, **Load_ab**, **max_a**, **max_b**, **dn**, **dne**, **Move_ab**, **Min_vls**, **Move** and **Load**, corresponds to a different function of the CA model implemented as a VHDL component. **Check** block is the main component for checking of the states of the cells ahead in order to provide the critical details for possible obstacle, i.e. another train, station, or station with train; while the **Load_ab** is responsible for the acceleration/deceleration calculation taking into account the train velocity and leading the corresponding blocks **max_a** and **max_b**, respectively, as indicated by the earlier described CA model pseudocode. The components **dn** and **dne** are responsible for the calculation of the minimum single instantaneous distance and the distance resulting from the calculations and the considered cases described in eq. 1. Finally, blocks **Move** and **Move_ab** are related with the train movement and certain checks for safety reasons, while the

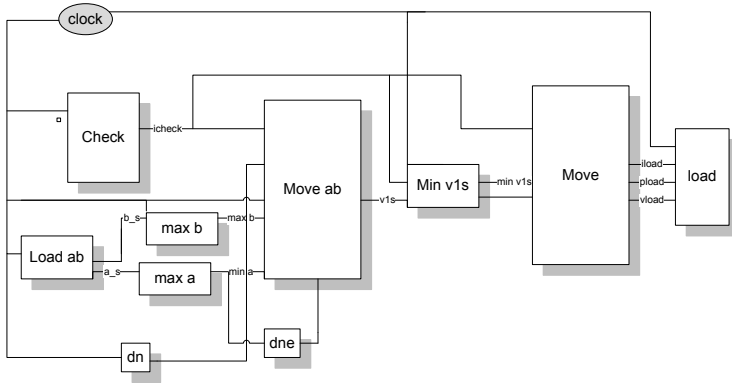


Fig. 2. CA cell block diagram

`Min_v1s` component acts as a safety valve which is able to handle possible malfunctions on the train velocity in conjunction with the calculated distance. The `clock` signal provides the essential synchronization to the above units.

In order to generate the whole CA system, some extra VHDL components are used for the interconnection of the CA cells as well as for handling of the aforementioned boundary conditions. More specifically, a semi parallel module inspired by serpentine memory and responsible for loading the initial conditions for each CA cell has been preferred resulting on a reduced number of pins. On the other hand, the overall system frequency has been also diminished. Regarding the boundary conditions special CA cells with fewer components and zero initial load conditions, thus empty cells, have been properly added to produce the prospective results. It should be also mentioned that in order to escalate the robustness and the adaptability of the proposed design a special compilation function was developed in order to automatically generate the VHDL code of the CA network model design, has been developed. In particular, this translation function receives the programming code of the CA model originally written in Matlab as its input, and automatically produces, as output, the corresponding synthesizable VHDL code. To achieve its goal, the translation function collects information from the presented CA model by checking its primary parameters. More specifically, the train network topology is used to produce the interface and the behavioral parts of the VHDL code, whereas the trains network boundary conditions and the initial train network parameters' values are used to produce the structural parts of the VHDL code. No previous knowledge of VHDL is required, since the VHDL code is directly produced from the high-level programming language code through the translation algorithm. However, as shown in Fig. 4 there is always a possibility of functional simulation of the VHDL code with the use of the appropriate automatically generated test benches. Finally, the automatically produced synthesizable VHDL CA code is translated into a hardware schematic of the defined architecture using predetermined timing constraints in Quartus II, v. 7.2 design software. Design of the proposed processor results in an ALTERA Stratix EP1S25F1020C5 FPGA device, which indicates

a maximum clock rate around 110MHz, consists of 200 CA cells and uses 84% of the total available logic elements. For readability reasons and in order to get the operational principles of the CA model illuminated acutely, straightforward simulations have been chosen. Nevertheless, all major circumstances, regarding trains and stations are included. More specifically, the time-space diagrams of train traffic flow for $v_{max} = 10$, $L = 30$, $a = 1$, $b = 1$, and $t_d = 1$, as well as the corresponding functional simulation screens of the resulting FPGA are all presented in Fig. 3 and Fig. 4, respectively. Consequently, the FPGA simulation results are found in complete agreement with the compilation results of the CA model.

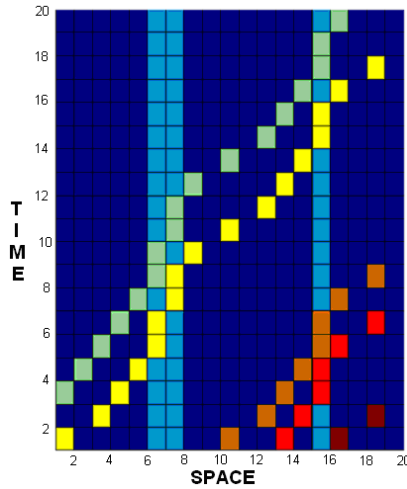


Fig. 3. Time-space diagrams of train movement for $v_{max} = 10$, $L = 30$, $\alpha = 1$, $b = 1$, and $t_d = 1$. The stations of different size are depicted with light blue, while all other colours are used to represent different moving trains.

4 GPS Integration

The aforementioned FPGA design could maximize its performance with the help of a global positioning system (GPS) and wireless communication module installed all in one system in each train and every station, monitor all automotive activity across the railway network. In such a way the tracking of the train movement as well as its velocity calculation will be straightforward. As a result, a portable, small, high-performance wireless device has been designed as depicted in Fig. 5. The aforementioned circuit is fully programmable using a microcontroller PIC 16F877 and RS-232 interconnection and provides the ability to collect and transfer both analog and digital data. Beyond its small size, it can work in any environmental condition (from -55 to 125° Celsius) and operate both as a server and answer on-demand to analog and digital call, while it can change the operation to a

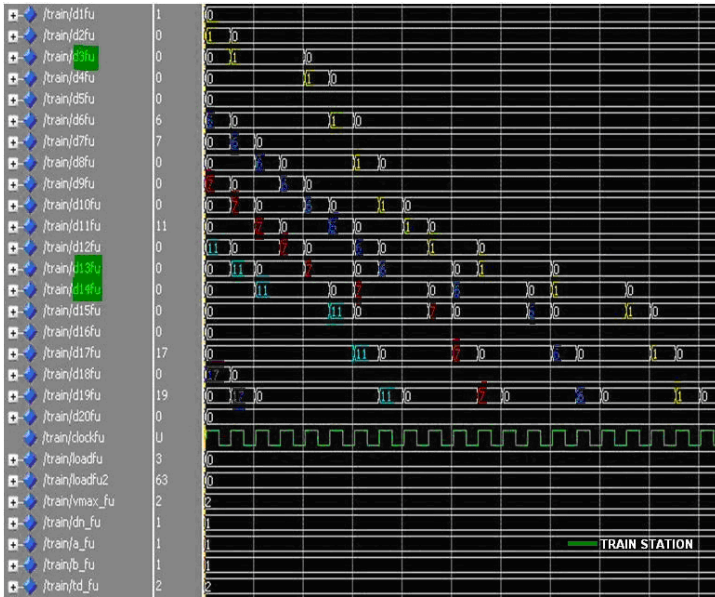


Fig. 4. The corresponding functional simulation screens of the resulting FPGA in accordance with the time-space diagrams of Fig. 3.

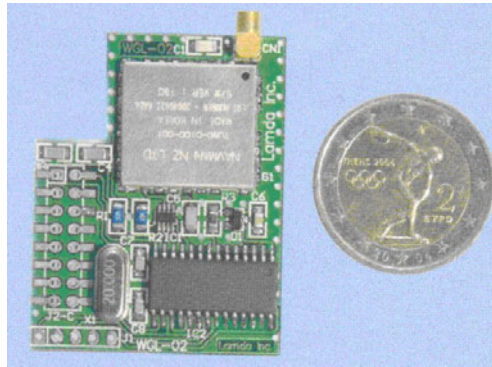


Fig. 5. The proposed small portable high performance GPS wireless communication module

client and send the requested information in accordance to FPGA circuit demand. Furthermore, there are approximately 17 measurements that can be transmitted over kilobyte with this module, providing low-cost operation as well. Thanks to these features and flexibility, the aforementioned module in conjunction with the proposed fully automatically programmable FPGA device minimizes the design burden offering the chance of a fully autonomous portable electronic system able to provide train control operation based on the presented CA model.

In case of GPS communication the NMEA communication protocol has been used. The NMEA 0183 although it is an old protocol compared to SIRF or other new GPS protocols and rather slow since its operation function is reduced to 4800 baud it is preferred for the 95% of the GPS/tracking devices. The main reason is that the data sent through the device are few and as a result even such low performances are not prohibitive for the proper functionality of the proposed circuit. The train or/and station position can be found at any time through the GPS satellite network while the train velocity can be calculated from the fundamental relationship $v = s/t$ for two different time moments. The above calculation is succeeded in real time by the following equation:

$$s = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (2)$$

where x_1, y_1, z_1 are coordinates of the first measurement and x_2, y_2, z_2 are coordinates of the second measurement, in correspondence. For the x and y GPS coordinates the EEP WGS84 is used, where WGS84 is the world geodetic system of year 1984, and z is calculated as the absolute elevation from the sea level as specified by geoid.

5 Conclusions

In this paper a CA model for the simulation of railway traffic flow was introduced. The proposed model was inspired by Li et al. model [6] and aimed at efficient simulation of railway traffic while providing simplicity, adaptability, low complexity and near optimum usage of computational resources. The presented space-time diagrams of railway traffic flow and the trajectories of the train movement reproduce some nonlinear real train traffic phenomena as found in the previous works. In order to leverage the natural parallelism of CAs, the presented model was automatically implemented with the help of VHDL code on a FPGA device. The resulting FPGA design is found promising in terms of computational performance, portability and low power consumption. Furthermore it can be easily connected to a small portable high performance GPS wireless communication module proposed in this paper. Consequently, the resulted FPGA processor is fed with real data about the trains location and speed and feedbacks with a support decision system providing on time information regarding the possible optimal control of the railway network. As future work concerns, the expansion of the CA model for handling more complicated situations, as for example, different track geometries, different signalling, multiple lanes, more than one station platforms, different and probably longer station dwell times so as the trains to pick up large crowds of waiting passengers, etc. should be considered. At this point, some real case experiments will be taking place in the railway network of Eastern Makedonia-Thrace of Greece and the experimental results will validate the calibration of the CA model as well as of the proposed electronic system.

Acknowledgments. The authors would like to thank Lamdas (λ) Electronics Inc. for their valuable assistance in design and development of the proposed GPS wireless module.

References

1. Evans, A.W.: Are Train Accident Risks Increasing? *Modern Railways* 59(647), 49–51 (2002)
2. Gill, D.C., Goodman, C.J.: Computer-based optimization techniques for mass transit railway signalling design. *IEE Proc.-B* 139(3), 261–275 (1992)
3. Ho, K., Wong, K.K.: Peak power demand reduction under moving block signalling using an expert system. *IEE Proc. Electr. Power Appl.* 150(4), 471–482 (2003)
4. Liy, R., Golovitcher, I.M.: Energy-efficient operation of rail vehicles. *Transp. Res. A* 37(10), 917–932 (2003)
5. Fu, Y., Ziyou, G., Li, K.: Modeling Study for Tracking Operation of Subway Trains Based on Cellular Automata. *J. Transpn. Sys. Eng. and IT* 8(4), 89–95 (2008)
6. Li, K.P., Gao, Z.Y., Ning, B.: Cellular automaton model for railwayway traffic. *Journal of Computational Physics* 209(1), 179–192 (2005)
7. Li, K.P., Gao, Z.Y., Ning, B.: Modeling the railway traffic using cellular automation model. *Inter. J. Mod. Phys. C* 16(6), 921–932 (2005)
8. Nagel, K., Schreckenberg, M.: A Cellular automaton model for freeway traffic. *J. Phys. I* 2(12), 2221–2229 (1992)
9. Xun, J., Ning, B., Li, K.-P.: Station Model for Rail Transit System Using Cellular Automata. *Commun. Theor. Phys.* 51, 595–599 (2009)
10. Spyropoulou, I.: Modelling a signal controlled traffic stream using cellular automata. *Transportation Research Part C* 15, 175–190 (2007)
11. Tomoeda, A., Komatsu, M., Yoo, I.Y., Uchida, M., Takayama, R., Nishinari, K.: Real-Time Railway Network Simulator “KUTTY”. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008. LNCS*, vol. 5191, pp. 433–440. Springer, Heidelberg (2008)
12. Mardiris, V., Sirakoulis, G.C., Mizas, C., Karafyllidis, I., Thanailakis, A.: A CAD system for modeling and Simulation of Computer Networks using Cellular Automata. *IEEE Transactions on Systems, Man and Cybernetics, Part C* 38(2), 253–264 (2008)
13. Tomoeda, A., Nishinari, K., Chowdhury, D., Schadschneider, A.: An information-based traffic control in a public conveyance system: Reduced clustering and enhanced efficiency. *Physica A: Statistical Mechanics and its Applications* 384(2), 600–612 (2007)

What Do We Mean by Asynchronous CA? A Reflection on Types and Effects of Asynchronicity

Stefania Bandini, Andrea Bonomi, and Giuseppe Vizzari

Complex Systems and Artificial Intelligence (CSAI) research center
Department of Computer Science, Systems and Communication (DISCo)
University of Milan - Bicocca
Viale Sarca 336/14, 20126 Milano, Italy
{bandini, bonomi, vizzari}@disco.unimib.it

Abstract. The aim of this paper is to introduce the problematics deriving from the adoption of an asynchronous CA model. First of all, several cellular automata update schemes and a tentative classification of such schemes are introduced. In order to study the effects of the different update schemes, we introduced a class of simple CA, called One Neighbor Binary Cellular Automata (1nCA). An overview of the general features of 1nCA is described, then the effects of six different updates schemes on all the class of 1nCA are described.

1 Introduction

Cellular Automata have traditionally treated time as discrete and state updates as occurring synchronously and in parallel. However, several authors [1,2] have argued that asynchronous models are viable alternatives to synchronous ones and suggest that asynchronous models should be preferred where there is no evidence of a global clock in the modeled reality.

There are several asynchronous cellular automata update schemes. [3] introduced an asynchronous model characterized by different cell updating schemes, basically sequential ones, in which a single cell is updated at each time step. The order of the updating sequence is defined as one of the following three methods: *Random order*, *Fixed Random Order*, and *Interlaced order*.

In [4] three classes of update scheme are identified: Synchronous Update, Random Asynchronous (RAS), and Ordered Asynchronous (OAS). The first scheme is the traditional CA updating scheme; according to the Random Asynchronous scheme, at any given time individuals to be updated are selected at random according to some probability distribution. In the Ordered Asynchronous update process, the updating of individual states follows a systematic pattern. The authors consider a total of six update patterns, including two RAS schemes and three OAS scheme: *Synchronous Scheme*, *Random Independent (RAS)*, *Random Order (RAS)*, *Cyclic (OAS)*, *Clocked (OAS)*, and *Self-Sync*. The author chose to implement local synchrony by using a coupled oscillator approach. The period of each timer is adjusted after an update so as to more closely match the period of other cells in its neighborhood.

The aim of this paper is to provide a comprehensive analysis of different asynchronous update schemes and to evaluate the effect of their adoption in a simplified CA

model; the paper is organized as follows: the following Section formally introduces a comprehensive set of relevant updating schemes, while Section 3 introduces the model in which the different update schemes will be tested. Section 4 describes the effects of the adoption of the different update schemes for this model, while conclusions and future developments end the paper.

2 A Classification of Update Schemes

In order to classify the update schemes, we define the following parameters:

- $p_i^{(t)}$ determines the period of the update of the cell i at the time step t , i.e. how many time steps the cell i will wait in order to be updated. The value of p can change during the time, e.g. in the Self-Sync update scheme.
- $l_i^{(t)}$ determines the length (in terms of time step) of the updating of the cell i at the time step t , i.e. after how many time steps the neighbor cells taking into account the new state during their updated.
- d_i , determines the delay (in terms of time step) before the first update.
- $U^{(t)}$ is the set of cells beginning the update process at the at the time step t .
- $u^{(t)} = |U^{(t)}|$ is the number of cells starting the update process at the time step t .

Given the above parameter, a set of relevant update schemes will now be presented and discussed. For each update scheme, we give a formal definition that are successively employed for the classification of the update schemes.

2.1 Relevant Update Schemes

Synchronous Scheme – All individuals are updated in parallel at each time step. The updating of a cell takes 1 time step.

$$\forall t \in \mathbb{Z} t > 0 \quad \forall i \in \mathbb{Z} \quad 0 \leq i < N \quad p_i^{(t)} = 1 \quad l_i^{(t)} = 1 \quad d_i = 0 \quad u^{(t)} = N$$

Random Independent – At each time step, one and only cell, chosen at random, is updated. The updating of a cell takes 1 time step.

$$\forall t \in \mathbb{Z}, t > 0 \quad \forall i \in \mathbb{Z} \quad 0 \leq i < N \quad l_i^{(t)} = 1 \quad u^{(t)} = 1 \quad \exists t, i \quad p_i^{(t)} > 1$$

Random Order – All nodes are updated in random order. After the updating off all the nodes, the order is changed. The updating of a cell takes 1 time step. The maximum length of the update period is less than $2N$.

$$\forall t \in \mathbb{Z} \quad t > 0 \quad \forall i \in \mathbb{Z} \quad 0 \leq i < N \quad p_i^{(t)} < 2N \quad l_i^{(t)} = 1 \quad d_i < N \quad u^{(t)} = 1$$

We can define an update interval $[\alpha, \omega]$ so that

$$\forall z \in \mathbb{Z}, z > 0 \quad \alpha = 1 + z N \quad \omega = (z + 1) N$$

In every update interval, each cell is update exactly one time:

$$\forall i \in \mathbb{Z} \quad 0 \leq i < N, \quad \forall t_n \in \mathbb{Z}, \alpha \leq t_n \leq \omega, \quad \forall t_m \in \mathbb{Z}, \alpha \leq t_m \leq \omega, \\ c_i \in U^{(t_n)}, c_i \in U^{(t_m)} \iff t_n = t_m$$

Cyclic – At each time step a node is chosen according to a fixed update order.

$$\forall t \in \mathbb{Z} \quad t > 0 \quad \forall i \in \mathbb{Z} \quad 0 \leq i < N \quad p_i^{(t)} = N \quad l_i^{(t)} = 1 \quad d_i < N \quad u^{(t)} = 1$$

We can identify three subtypes of this update scheme:

Random Cyclic – The update order is decided at random during initialisation of the automaton. This update scheme correspond to the Kannada’s *Fixed Random* [3] and Cornforth’s *Cyclic OAS* [4].

Fixed Cyclic-Sequential Ordered – The update order is fixed in the automaton definition. The cells are updated one-by-one according to their natural order:

$$d_i = 1 + i; \quad q^t = (t - 1) \bmod N; \quad u^{(t)} = \{c_{q^t}\}.$$

Fixed Cyclic-Interlaced Cyclic – Called *Interlaced Order* in [3]. The set of cell $u^{(t)}$ to be update at time step t is calculated as $q^t = C(t - 1) \bmod N$; $u^{(t)} = \{c_{q^t}\}$, where C is a parameter prime to N .

Generic Cyclic – It is a generalization of the Cyclic update scheme, obtained relaxing the constraint on the updating length. In this update scheme, the updating length is limited only by the period. Formally:

$$\forall t \in \mathbb{Z} \ t > 0 \quad \forall i \in \mathbb{Z} \ 0 \leq i < N \quad p_i^{(t)} = N \quad l_i^{(t)} \leq p^{(t)} \quad d_i < N \quad u^{(t)} = 1$$

Clocked – A *timer* is assigned to each cell, so that updating is autonomous and proceeds at different rates for different cells. The update frequency of each cells is fixed:

$$\forall t \in \mathbb{Z} \ t > 0 \quad \forall i \in \mathbb{Z} \ 0 \leq i < N \quad p_i^{(t)} = p_i^{(0)} \quad l_i^{(t)} \leq p_i^{(0)} \quad d_i \leq p_i^{(0)}$$

As subtype of the Clocked update scheme is the *Equal Frequency Clocked*. According to this update scheme, every cells has the same update frequency:

$$\forall t \in \mathbb{Z} \ t > 0 \quad \forall i \in \mathbb{Z} \ 0 \leq i < N \quad p_i^{(t)} = p_0^{(0)}$$

Generic Clocked – It is a generalization of the Cyclic update scheme, obtained relaxing the constraint on the fixed update frequency. The two subtypes of this update scheme are the *Clocked* and *Variable Clocked*. According to the Variable Clocked scheme, a timer is assigned to each cell, so that updating is autonomous and proceeds at different rates for different cells. The updating frequency is not fixed: $\exists t, i : p_i^{(t)} \neq p_i^{(0)}$. The *Self-Sync* update scheme is an example of Variable Clocked scheme.

3 One Neighbor Binary Cellular Automata

One Neighbor Binary Cellular Automata (1nCA) is a one-dimensional Cellular Automata, with two possible states per cell. Each cell has two neighbors, left and right, defined to be the adjacent cells on either side, but the update rule consider only one neighbor per step. The neighborhood includes the cell itself and the left or the right adjacent cell and alternates between these two situations at even and odd time steps.

The size of the neighborhood is always 2, so there are 4 possible patterns for the neighborhood and only 16 possible rules. The number of possible rules is small compared to the 256 possible rules of the Elementary Cellular Automata, so it is easier to exhaustively study the dynamic behavior of the all rules.

These 16 1nCA rules will be referred using the Wolfram notation, with the rule numbers followed by the Δ symbol to avoid confusion with the Elementary Cellular Automata rules (e.g. “Rule 10” is an Elementary Cellular Automata rule, “Rule 10 Δ ” is an 1nCA rule).

We call One Neighbor Binary Cellular Automata the cellular automata $(\mathcal{L}, \mathcal{S}, \mathcal{N}, \mathcal{F})$ where

- $\mathcal{L} = [c_0, c_1, \dots, c_n]$ is an array of n cells,
- $\mathcal{S} = \{0, 1\}$ is the set of states ($k = 2$),
- \mathcal{N}_c is neighborhood of the cell c and $\forall c \in \mathcal{L} \quad |\mathcal{N}_c| = 2$,
- $f : \mathcal{S}^2 \rightarrow \mathcal{S}$ is a transition function.

Denoting the cell c at position i as c_i , the neighborhood $\mathcal{N}_{c_i}^{(t)}$ of the cell c_i at time t is defined as $\mathcal{N}_{c_i}^{(t)} = [c_i, n_{c_i}^{(t)}]$ where $n_{c_i}^{(t)}$ is the neighbor of the cell, given by

$$n_{c_i}^{(t)} = \begin{cases} c_{i+1} & \text{if } t \text{ is even} \\ c_{i-1} & \text{otherwise} \end{cases}$$

Following the Wolfram’s notation, the rules are characterized by a sequence of binary values ($\beta_i \in \mathcal{S}$) associated with each of the 4 possible patterns for the neighborhood. The transition function is defined as:

$$f(c_i, n_{c_i}^{(t)}) = \begin{cases} \beta_0 & \text{if } c_i = 0, n_{c_i}^{(t)} = 0 \\ \beta_1 & \text{if } c_i = 0, n_{c_i}^{(t)} = 1 \\ \beta_2 & \text{if } c_i = 1, n_{c_i}^{(t)} = 0 \\ \beta_3 & \text{if } c_i = 1, n_{c_i}^{(t)} = 1 \end{cases}$$

There are 16 possible transition functions, identified by a rule number $R = \sum_{i=0}^3 \beta_i 2^i$.

The configuration of a cellular automata is a mapping $q : \mathcal{L} \rightarrow \mathcal{S}$ which assigns to each cell of the array \mathcal{L} a state from \mathcal{S} . We denoted with q_t the configuration of a cellular automata at time t ; in particular $q_t = [s_0, s_1, \dots, s_n] \in \mathcal{S}^n$ (16) where n is the number of cells of \mathcal{L} . Given an initial configuration q_0 , the evolution of an automaton is represented by a sequence of configurations $q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow \dots \rightarrow q_t$. A deterministic finite cellular automaton eventually falls into a cycle (with period $p > 1$) or a fixed point ($p = 1$):

$$q_t \rightarrow q_{t+1} \rightarrow q_{t+2} \rightarrow \dots \rightarrow q_{t+p}$$

$$q_t = q_{t+p}; \quad q_{t+1} = q_{t+p+1}; \quad q_{t+2} = q_{t+p+2}; \quad \dots; \quad q_{t+p} = q_{t+2p}.$$

We defined two constant configurations $\bar{0}$ and $\bar{1}$ as: $\bar{0} = [0, 0, \dots, 0] \in \mathcal{S}^n$; $\bar{1} = [1, 1, \dots, 1] \in \mathcal{S}^n$.

In the following section a classification of the InCA Rules is presented. A central issue in the theory of cellular automata is the classification, i.e. understanding how cellular automata can be meaningfully grouped according to their structure and behavior. There are mainly two approach for the classification of the cellular automata: the direct way, called Phenotypic Classification, to classified cellular automata is to observe their behavior through the spatial-temporal patterns they generates out of several random initial conditions, and then to use statistical metrics to quantify the observed behavior [5]. Another approach, called Genotypic Classification, is based on the analysis of the automaton transition rules.

There are several works (e.g. [6,7,8,9,10]) focusing on the classification of the one dimensional cellular automata and in particular on the Elementary Cellular Automata. In this section we present an approach of genotypic classification applied to the One Neighbor Binary Cellular Automata. The idea of a genotypic classification of cellular automata is to divide a population of automata into groups according to the intrinsic properties of the rules. The aim is that some features of the cellular automata behaviors are predictable on the basis of a genotypic classification.

3.1 Totalistic Rules

A cellular automaton is called *totalistic* if the value of a cell depends only on the sum of the values of its neighbors at the previous time step, and not on their individual values [11]. Therefore, half of the possible rules for 1nCA are *totalistic*. The sum n of the neighborhood cells is computed $n = c_i + n_{c_i}^{(t)}$ and $0 \leq n \leq 2$. The following rules are totalistic:

$$\begin{aligned}
 \text{Rule } 0 \triangle f(n) &= 0 & \text{Rule } 8 \triangle f(n) &= \begin{cases} 0 & \text{if } n = 0 \\ 0 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \end{cases} \\
 \text{Rule } 1 \triangle f(n) &= \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n = 1 \\ 0 & \text{if } n = 2 \end{cases} & \text{Rule } 9 \triangle f(n) &= \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \end{cases} \\
 \text{Rule } 6 \triangle f(n) &= \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ 0 & \text{if } n = 2 \end{cases} & \text{Rule } 14 \triangle f(n) &= \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \end{cases} \\
 \text{Rule } 7 \triangle f(n) &= \begin{cases} 1 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ 0 & \text{if } n = 2 \end{cases} & \text{Rule } 15 \triangle f(n) &= 1
 \end{aligned}$$

3.2 Neighbor-Independent and Self-Independent

A rule is *Neighbor-Independent* if the value of a cell depends only on its previous value and not on the value of the neighbors. Formally, a rule is *Neighbor-Independent* if $\forall s \in \mathcal{S}, f(s, 0) = f(s, 1)$ so, according to the definition of the transaction function, a rule is *Neighbor-Independent* if $\beta_0 = \beta_1, \beta_2 = \beta_3$.

A rule is *Self-Independent* if the value of a cell depends only on the value of the neighbors and not on its previous value. Formally, a rule is *Self-Independent* if $\forall s \in \mathcal{S}, f(0, s) = f(1, s)$ so, according to the definition of the transaction function, a rule is *Self-Independent* if $\beta_0 = \beta_2, \beta_1 = \beta_3$.

3.3 λ -parameter

An even cruder piece of information about a rule is the number of non-quiescent outputs in a rule-table. For the One Neighbor Binary Cellular Automata, this parameter is equal to the number of β parameters that are equal to one and can be calculated as $c = \sum_{i=0}^3 \beta_i$.

Langton [12] proposed the so called λ -parameter as an order-chaos parameter for Cellular Automata. This parameter measures the density of non-quiescent (not zero) outputs in a rule-table. For the One Neighbor Binary Cellular Automata the λ -parameter can be calculated as: $\lambda = \frac{c}{k^n} = \frac{1}{4} \sum_{i=0}^3 \beta_i$ where k is the number of states and n is the neighborhood size. λ varies between 0 (order) to 0.5 (chaos) to 1 (order). As λ is

increased from 0 to 0.5 (or decreased from 1 to 0.5), the automata move from having the most homogeneous rule tables to having the most heterogeneous.

Langton presented evidence that there is some correlation between the λ parameter and the behavior of an “average” Cellular Automata on an “average” initial configuration [12]. Behavior was characterized in terms of quantities such as single-site entropy, two-site mutual information, difference-pattern spreading rate, and average transient length. Generally the correlation is quite good for very low and very high λ values, which predict fixed-point or short-period behavior. However, for intermediate λ values, there is a large degree of variation in behavior [13].

3.4 Sensitivity

[14][15] proposed the *sensitivity parameter* μ , motivated by the observation that the Wolfram classes are characterized by its sensitivity to changes in the state of a unique cell of the neighborhood of the transition rule.

Sensitivity is defined as the number of changes in the outputs of the transition rule, caused by changing the state of each cell of the neighborhood, one cell at a time, over all possible neighborhoods of the rule being considered: $\mu = \frac{1}{nm} \sum_n \sum_{j=1}^m \frac{\delta f}{\delta s_j}$ where m is

the number of cells in the neighborhood and n is the number of possible neighborhoods in the rule table. For 1 Neighbor Cellular Automata, $m = 2$, and $n = 2^m = 4$. The Boolean derivate for Cellular Automata [16] $\frac{\delta f}{\delta s_j}$ is equal to 1 if $f(s_1, \dots, s_j, \dots) \neq f(s_1, \dots, \neg s_j, \dots)$, otherwise is equal to 0.

The sensitivity parameter takes on three different values: 0, 0.5, and 1. The sensitivity parameter helps to relatively discriminate null and chaotic behaviors: the null behavior happens in rules with low sensitivity and the chaotic behavior happens in rules with high sensitivity. Fixed-point and periodic behaviors are concentrated around 0.5.

3.5 Rule Density

The Rule density is a simply parameter introduced to describe the rules behavior. The rule density, $R\rho$, is computed as $R\rho = (\lambda - \frac{1}{2}) 2^{(\beta_3 - \beta_0)} + \frac{1}{2}$.

Roughly speaking, rule density indicates the average fraction of sites whose value is one in the rule dynamic evolution. The rule density value is comprised between zero and one. A value of zero indicated that a rule converges (for most of the initial configurations) to zero state in all the cells, a value of one indicated a convergence to one.

3.6 Rules Symmetries

One means of verification of the consistence of the rule density parameter (and also the other parameters) is the use of symmetries: if two rules are *conjugate*, the rule density of one rule is equals to $1 - R\rho$ of the other rule.

In [17] the author defines the reflected, conjugate and reflected conjugate symmetries for the Elementary Cellular Automata. The only possible symmetry for the 1nCA

Table 1. The rules divided according to the symmetries. The value of *rule density* is reported for each rule. The classes marked with **T** are formed by totalistic rules, **N** by Neighbor-Independent rules, and **S** by Self-Independent rules.

Class 0 Δ TNS :	Rule 0 Δ ($R\rho = 0$)	Rule 15 Δ ($R\rho = 1$)
Class 1 Δ T :	Rule 1 Δ ($R\rho = 0.375$)	Rule 7 Δ ($R\rho = 0.625$)
Class 2 Δ :	Rule 2 Δ ($R\rho = 0.25$)	Rule 11 Δ ($R\rho = 0.75$)
Class 3 Δ N :	Rule 3 Δ ($R\rho = 0.5$)	
Class 4 Δ :	Rule 4 Δ ($R\rho = 0.25$)	Rule 13 Δ ($R\rho = 0.75$)
Class 5 Δ S :	Rule 5 Δ ($R\rho = 0.5$)	
Class 6 Δ T :	Rule 6 Δ ($R\rho = 0.5$)	Rule 9 Δ ($R\rho = 0.5$)
Class 8 Δ T :	Rule 8 Δ ($R\rho = 0$)	Rule 14 Δ ($R\rho = 1$)
Class 10 Δ S :	Rule 10 Δ ($R\rho = 0.5$)	
Class 12 Δ N :	Rule 12 Δ ($R\rho = 0.5$)	

is when f^* , the conjugate rule of f , is defined as $\forall(c_i, n_{c_i}) \in \mathcal{S}^2, f^*(c_i, n_{c_i}) = f(\neg c_l, \neg n_{c_i})$ where \neg denotes the operation of changing the zeros into ones and ones into zeros. The β^* parameters of the conjugate rule are defined as $\beta_3^* = \neg\beta_0; \beta_2^* = \neg\beta_1; \beta_1^* = \neg\beta_2; \beta_0^* = \neg\beta_3$.

We identified 6 classes of rules, shown in Table 1, according to the symmetries: we can group in one class all the rules that are symmetric (reflected, conjugated or reflected conjugated). The classes are named according to the lowest member index. Each class is formed by totalistic or non-totalistic rules.

This kind of classification of the One Neighbor Binary Cellular Automata is important because we can restrict the study of the dynamic behavior to only one member of each class and the behavior of the other members can be simply inferred according to the symmetric relations.

4 InCA Spatiotemporal Patterns

In this section we present the effects of several update schemes on InCA automata dynamic evolutions. In Figure 2 the time evolutions of all the 16 rules according with synchronous update scheme and periodic boundaries conditions, starting from an initial random configuration of 60 cells. As shown in the following sections, the different update schemes produce tremendous effects on the several automata.

The tested the following update schemes on all the InCA rules:

- Synchronous
- Random Cyclic
- Equal Frequency Clocked
- Random Order
- Random Independent

4.1 Class 6 Δ T

In this paragraph, we present the observation results for the Class 6 Δ , as an example of dynamic evolution. The rules of this class are Rule 6 Δ ($R\rho = 0.5, \lambda = 0.5, \mu = 1$) and Rule 9 Δ ($R\rho = 0.5, \lambda = 0.5, \mu = 1$). The rules of this class are *Chaotic*: these rules

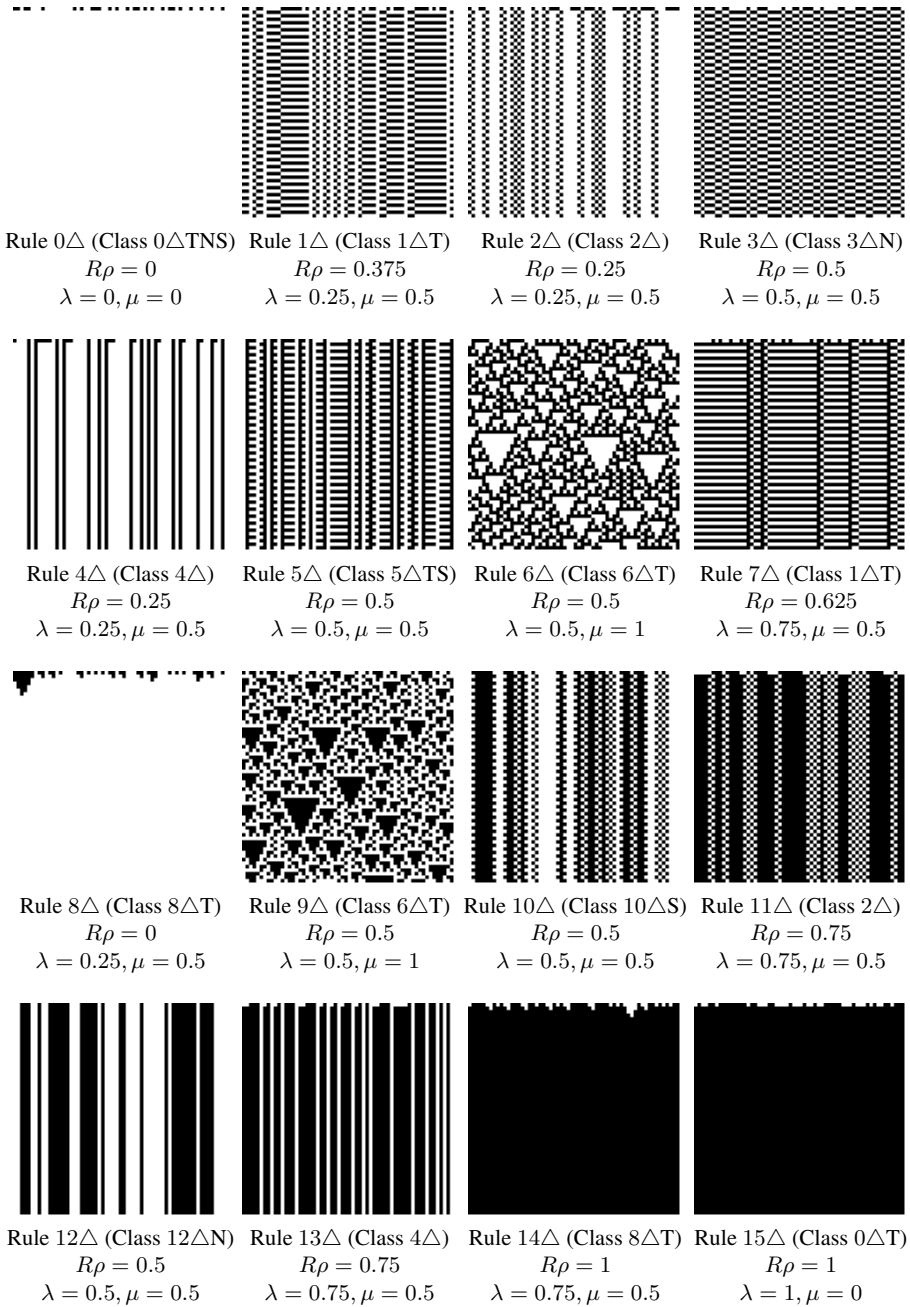


Fig. 1. 60 steps of the time evolution of all the 16 One Neighbor Binary Cellular Automata with the default synchronous update scheme and periodic boundaries conditions starting from an initial random configuration of 60 cells

Table 2. Classification summary

Rule	Bin	Wolfram Class	Li-Packard Class	Symmetry Class	Tot. Ind.	λ	$R\rho$	μ	
0 Δ	0000	W1	Null	0 Δ	T	NS	0	0	0
1 Δ	0001	W2	Two-Cycle	1 Δ	T		0.25	0.375	0.5
2 Δ	0010	W2	Two-Cycle	2 Δ			0.25	0.25	0.5
3 Δ	0011	W2	Two-Cycle	3 Δ		N	0.5	0.5	0.5
4 Δ	0100	W1	Fixed-Point	4 Δ			0.25	0.25	0.5
5 Δ	0101	W2	Two-Cycle	5 Δ		S	0.5	0.5	0.5
6 Δ	0110	W3	Chaotic	6 Δ	T		0.5	0.5	1
7 Δ	0111	W2	Two-Cycle	1 Δ	T		0.75	0.625	0.5
8 Δ	1000	W1	Null	8 Δ	T		0.25	0	0.5
9 Δ	1001	W3	Chaotic	6 Δ	T		0.5	0.5	1
10 Δ	1010	W2	Two-Cycle	10 Δ		S	0.5	0.5	0.5
11 Δ	1011	W2	Two-Cycle	2 Δ			0.75	0.75	0.5
12 Δ	1100	W2	Fixed-Point	12 Δ		N	0.5	0.5	0.5
13 Δ	1101	W2	Fixed-Point	4 Δ			0.75	0.75	0.5
14 Δ	1110	W1	Null	8 Δ	T		0.75	1	0.5
15 Δ	1111	W1	Null	0 Δ	T	NS	1	1	0

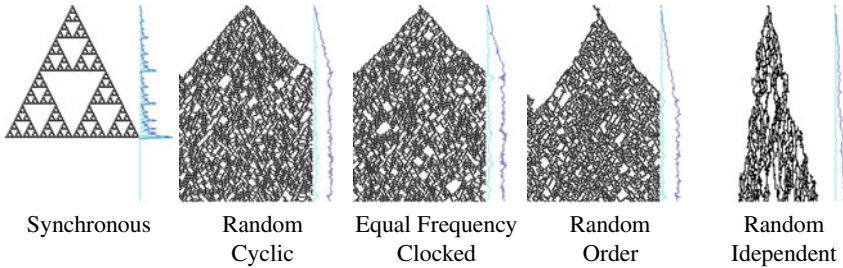


Fig. 2. Time space diagrams of Rule 6 Δ using different update schemes starting from a single seed

are characterized by the exponential divergence of its cycle length with the system size, and for the instability with respect to perturbations. If the number of cells is finite, for the Synchronous, Random Cyclic, and Equal Frequency Clocked schemes, the evolution eventually falls into a cycle (with period $p > 1$) or a fixed point ($p = 1$). The configuration $\bar{0}$ is the fixed point of the Rule 6 Δ , the configuration $\bar{1}$ is the fixed point of the Rule 9 Δ . These configurations are fixed points also using the Random update schemes.

Changing update scheme has dramatic effect on the rule of this class. As shown in Figure 2 with the Synchronous update scheme, the Rule 6 Δ produces a dynamic evolution similar to the *Sierpinski Triangle* fractal. This typical shape is not present with any of the other update schemes.

Moreover if the automaton has periodic boundaries conditions and the number of cells is a power of two, starting from an initial configuration, the evolution of the synchronous automata eventually reaches the fixed point. The automata with the other update schemes does not have this behavior.

5 Conclusions and Future Developments

The paper has presented a discussion on asynchronicity in CA models, comparing different types of update scheme and proposing an ontology to classify them. The implications of the different update schemes have been presented by introducing a very simple CA based model and testing it adopting different update schemes. Future developments of this work, in the vein of [18], are aimed at evaluating the possibility to define asynchronous models in which some global dynamic properties are preserved even adopting different asynchronous update schemes.

References

1. Paolo, E.A.D.: Searching for rhythms in asynchronous random boolean networks. In: Bedau, M. (ed.) *Alife VII: Proc. of the 7th International Conference*, pp. 73–80. MIT Press, Cambridge (2000)
2. Thomas, R., European Molecular Biology Organization: *Kinetic Logic: a Boolean Approach to the Analysis of Complex Regulatory Systems*. Lecture notes in Biomathematics, vol. 29. Springer, Berlin (1979)
3. Kanada, Y.: The effects of randomness in asynchronous 1d cellular automata (poster). *Artificial Life IV* (1994)
4. Cornforth, D., Green, D.G., Newth, D.: Ordered asynchronous processes in multi-agent systems. *Physica D: Nonlinear Phenomena* 204(1-2), 70–82 (2005)
5. Li, W., Packard, N., Langton, C.G.: Transition phenomena in CA rule space. *Physica D* 45, 77 (1990)
6. Wolfram, S.: Cellular automata. *Los Alamos Science* 9, 2–21 (Fall 1983)
7. Gutowitz, H., Victor, J.D., Knight, B.W.: Local structure theory for cellular automata. *Physica D* 28, 18–48 (1987)
8. Li, W., Packard, N.: The structure of the elementary cellular automata rule space. *Complex Systems* 4(3), 281–297 (1990)
9. Sutner, K.: Classifying circular CA. *Physica D* 45, 386 (1990)
10. Wuensche, A.: Classifying cellular automata automatically: Finding gliders, filtering, and relating space-time patterns, attractor basins, and the Z parameter. *Complexity* 4(3), 47–66 (1999)
11. Wolfram, S.: Statistical mechanics of cellular automata. *Reviews of Modern Physics* 55, 601–644 (1983)
12. Langton, C.G.: Computation at the edge of chaos. *Physica D* 42, 12–37 (1990)
13. Mitchell, M., Hraber, P.T., Crutchfield, J.P.: Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems* 7, 89–130 (1993)
14. Binder, P.: Parametric ordering of complex systems. *Physical Review E* 49(3), 2023–2025 (1994)
15. Binder, P.: A phase diagram for elementary cellular automata. *Complex Systems* 7, 241–247 (1993)
16. Vichniac, G.Y.: Boolean derivatives on cellular automata. *Physica D* 45(1-3), 63–74 (1990)
17. Fatès, N.: Experimental study of elementary cellular automata dynamics using the density parameter. In: *Discrete Models for Complex Systems, DMCS 2003. DMTCS Proceedings, Discrete Mathematics and Theoretical Computer Science*, vol. AB, pp. 155–166 (2003)
18. Fatès, N., Morvan, M.: An experimental study of robustness to asynchronism for elementary cellular automata. *Complex Systems* 16(1), 1–27 (2005)

Parallel Composition of Asynchronous Cellular Automata Simulating Reaction Diffusion Processes*

Olga Bandman

Supercomputer Software Department
ICM&MG, Siberian Branch, Russian Academy of Sciences
Pr. Lavrentieva, 6, Novosibirsk, 630090, Russia
`bandman@ssd.sccc.ru`

Abstract. A method of constructing asynchronous cellular automata (ACA) as a parallel composition of two interacting ACA is presented. The resulting ACA is intended to simulate a process with more than one species being involved in it. Two cases of such a composition are considered: (1) when one ACA is functioning independently affecting the evolution of the other, and (2) when both ACA evolve interacting at each iteration.

1 Introduction

In spite of the fact that physical-chemical spatial processes are asynchronous by nature, the majority of mathematical models intended for simulating them are synchronous. This contradiction exists both in traditional continuous mathematics and in cellular automata (CA). Three main reasons of synchronous CA predominance are as follows: 1) the original von-Neumann's CA is a synchronous one; 2) synchronous mode corresponds both to numerical analysis principle and to hardware mode of operation; 3) synchronous computation process is more deterministic, and, hence, easier for programming and parallel implementing.

Recently, the scientific interest has been moved towards nonlinear self organizing processes, where two or more species are involved. Most of them deal with synchronous CA, e.g. biological examples implicitly given in [1,2] and physical and chemical complex phenomena studied in [3]. To eliminate the discrepancy between synchronous models and asynchronous behavior of real life phenomena a method for constructing an ACA as a parallel composition of two (or more) ones, each for one species, is presented.

The paper contains a brief description of the method, and illustrates it by two examples: one way composition of pattern formation on heated surface (one-way composition), and prey predatory interactions (two-way interaction). The paper concludes by some considerations about the results and future work.

* Supported by (1)Presidium of Russian Academy of Sciences, Basic Research Program N 2 (2009), (2)Siberian Branch of Russian Academy of Sciences, SBRAS Interdisciplinary Project 32 (2009).

2 Parallel ACA Composition Method

In [4] a CA-model of complex phenomenon, where several species are involved, is classified as a parallel CA composition. It suggests functioning of n interacting CA, each simulating the evolution of a corresponding species. For clearness, and aiming at asynchronous CA investigation, the composition of two ACA \aleph_1 and \aleph_2 is further considered. Each ACA is determined by three sets: $\aleph_k = \langle A_k, X_k, \Theta_k \rangle$, $k = 1, 2$, where A_k is a *state alphabet*, X_k - is a *set of cell names*, and Θ_k is a *local operator*. The alphabets A_1 and A_2 may be different and of any type (Boolean, real, symbolic). Between $X_1 = \{x_{i_1}\}$, and $X_2 = \{x_{i_2}\}$, $i = 1, 2, \dots, M$, there exists an one-to-one correspondence ξ , such that $x_{i_2} = \xi(x_{i_1})$, and $x_{i_1} = \xi^{-1}(x_{i_2})$. For simplicity, in the expressions valid for both ACA components, indices indicating ACA numbers are further omitted. In the sets X_k , $k = 1, 2$, the following *templates* are defined: $T_k(x_i) = \{x_{j_k} : j_k = 1, \dots, q\}$, $T_k(x_i) \in X_k$, and $S_k(x_i) = \{x_{j_1} : j_1 = 1, \dots, s\} \cup \{x_{j_2} : j_2 = 1, \dots, m\}$, $S_k(x_i) \in X_1 \cup X_2$, x_{j_k} being cells in the close vicinity of x_{i_k} , $k = 1, 2$. Each cell x_i at any moment t is endowed with a state $v_i(t) \in A$. The array of all cells states $\Omega(t) = \{v_i(t) : i = 1, \dots, |X|\}$ is a *global configuration*.

The two ACA are functioning in parallel, Θ_1 being applied to the cells of Ω_1 , and Θ_2 - to the cells of Ω_2 . The application of Θ_k , $k = 1, 2$, to a cell $x_i \in X_k$ replaces the states in its neighborhood $T_k(x_i) \in X_k$ by the values v'_j , ($j = 1, \dots, q$) of *transition functions* $f_{k_j}(V_k(x_i))$, where $V_k(x_i)$ is a set of states in the cells of $S_k(x_i) \in X_1 \cup X_2$. Functioning of the composed ACA proceeds by repeating the following steps: (1) a cell name $x_i \in X_1$ is chosen randomly, (2) local operator Θ_1 is applied to the chosen cell, (3) a cell name $x_i \in X_2$ is chosen randomly, (4) local operator Θ_2 is applied to the chosen cell. The above steps are repeated until the simulation terminates either when it achieves a stable state or when the the prescribed number of iterations T is exhausted.

3 One Way Parallel Composition of Asynchronous CA

One-way composition suggests \aleph_1 and \aleph_2 playing different roles. Let $\aleph_1 = \langle A_1, X_1, \Theta_1 \rangle$ represent the process under investigation, and $\aleph_2 = \langle A_2, X_2, \Theta_2 \rangle$ operate autonomously playing a controlling role. Hence, $S_1(x_1) \in X_1 \cup X_2$, $S_2(x_2) \in X_2$.

Example 1. Pattern formation process on an unevenly heated surface is simulated by a composition of two ACA: $\aleph_v = \langle A_v, X_v, \theta_v \rangle$, and $\aleph_u = \langle A_u, X_u, \theta_u \rangle$, where $A_v = A_u = A = \{0, 1\}$ and $|X_v| = |X_u| = \{(i, j) : i, j = 0, \dots, 300\}$, satisfying (1). ACA \aleph_v simulates heat propagation acting as a *naive diffusion CA* given in [5]. Initial temperature distribution is represented as $\Omega_v(0)$ (Fig.1a). Θ_v performs the exchange of states between the cell (i, j) and one of its four neighbors (v_k) : $v_0 \leftrightarrow v_k$.

$$v'_0 = v_k, \quad \text{if } 0.25k < rand < 0.25(k + 1),$$

$$v'_k = \begin{cases} v_0 & \text{if } 0.25k < rand < 0.25(k + 1), \\ v_k & \text{otherwise.} \end{cases} \quad k = 0, \dots, 3. \quad (1)$$

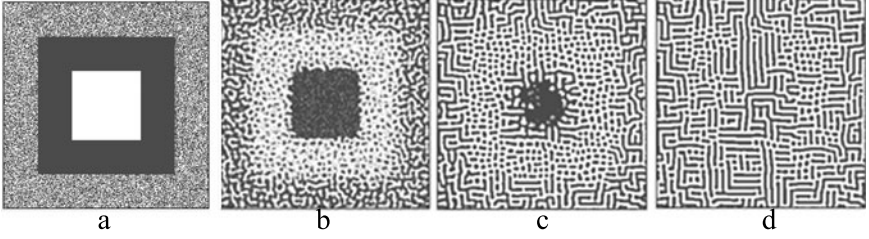


Fig. 1. Initial state $\Omega_v(0)$ – (a), and three snapshots of the evolution of \aleph_u with $t=2$ (b), $t=20$ (c), $t=150$ (d)

ACA \aleph_u , simulates pattern formation process. $\Omega_u(0)$ has randomly distributed "ones" with the mean density $\langle u \rangle = 0.5$. Θ_u changes the state of a cell (i, j) according to the transition function which depends on states of cells of $S_v(i, j) \cup S_u(i, j)$, where

$$S_v(i, j) = S_u(i, j) = \{(i + g, j + h) : g, h = -3, \dots, 3\}. \quad (2)$$

$$u'_0 = \begin{cases} 1, & \text{if } \sum_{g,h=-3}^3 w_{gh} u_{i+g,j+h} > 0.1, \\ 0, & \text{otherwise,} \end{cases}$$

$$w_{gh} = \begin{cases} 1, & \text{if } |g| \leq 1 \ \& \ |h| \leq 1, \\ -\langle v_{i+g,j+h} \rangle & \text{otherwise.} \end{cases}, \text{ where } \langle v_{\alpha,\beta} \rangle = (2r+1)^{-2} \sum_{a,b=-10}^{10} v_{\alpha+a,\beta+b}.$$

4 Two Way Parallel Composition of Asynchronous CA

A two-way composition is intended for simulation two interdependent processes. Hence, both ACA have local operators defined on subsets from both cellular arrays.

Example 2. On a certain area there are two species interpreted as predator and prey. If there is enough of prey for predatory to eat, predator density increases with the probability which depends on satiated predator density. In case of food shortage predator density diminishes. Prey always attempts to propagate. Both species diffuse. Predator being more agile is characterized by diffusion coefficient much larger than that of prey ($d_v \gg d_u$). ACA $\aleph_v = \langle A_v, X_v, \Theta_v \rangle$ stands for predator, $\aleph_u = \langle A_u, X_u, \Theta_u \rangle$ — for prey. Local operators Θ_v and Θ_u are sequential compositions of two local operators: 1) θ_{d_v} and θ_{d_u} simulate diffusion using transition function (1) which are applied with the probabilities $p_u(d_u)$ and $p_v(d_v)$, respectively; 2) θ_{b_v} and θ_{b_u} simulate predator and prey behavior. In their transition functions the templates $T_v(i, j) = (i, j)_v$, $T_u(i, j) = (i, j)_u$, $S_v(i, j)$ and $S_u(i, j)$ given by (2) are used.

The predator transition function computes the next state $v'_0(i, j)$ as follows:

$$v'_0(i, j) = \begin{cases} 0, & \text{if } U(i, j) > V(i, j) \ \& \ (rand) < p_{v \rightarrow 0}, \\ 1, & \text{if } V(i, j) > U(i, j) \ \& \ (rand) < p_{v \rightarrow 1}, \end{cases} \quad (3)$$

where $V(i, j)$ and $U(i, j)$ are sums of states in the cells of $S_v(i, j)$ and $S_u(i, j)$, respectively. The probabilities $p_{v \rightarrow 0}$ and $p_{v \rightarrow 1}$ are computed as follows

$$\begin{aligned} p_{v \rightarrow 0} &= (V(i, j) - U(i, j))/V(i, j), & \text{if } V(i, j) > U(i, j), \\ p_{v \rightarrow 1} &= 0.5V(i, j)/|S_v|(1 - V(i, j)/|S_v|), & \text{if } U(i, j) > V(i, j). \end{aligned}$$

The prey transition function of θ_{b_v} is similar to (3), differing in the probabilities $p_{u \rightarrow 0}$ and $p_{u \rightarrow 1}$.

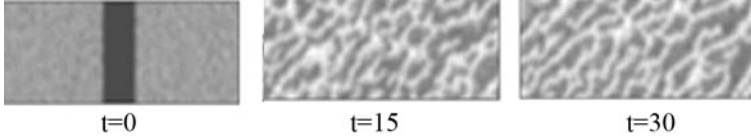


Fig. 2. Three snapshots of the evolution of the predator ACA

The important thing is that the composed ACA is very stable. All initial configurations having any area (sometimes very small) with nonzero species densities of both species tend to the same stable pattern.

5 Conclusions

Parallel composition method of asynchronous cellular automata is presented. Two particular cases are considered in detail and experimentally tested: one-way composition allows to introduce a controlling effect in the evolution of a process under simulation; two-way composition allows to construct ACA, simulating self organizing behavior. The results allow to hope that the approach might be helpful to create some techniques for constructing ACA when some behavioral properties of its evolution are given.

References

1. Deutsch, A., Dorman, S.: Cellular Automata Modeling of Biological Pattern Formation. Birkhäuser, Basel (2005)
2. Cattaneo, G., Dennunzio, A., Farina, F.: A Full Cellular Automaton to Simulate Predatory-Prey Systems. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 446–451. Springer, Heidelberg (2006)
3. Winzel, K.: Phase Transition of Cellular Automata Zeitschrift Für Physik. Condensed Matter 58(3), 224–229 (1985)
4. Bandman, O.: Cellular Automata Composition Techniques for Spatial Automata Simulation. In: Kroc, J., Sloot, P.M.A. (eds.) Simulating Complex Systems by Cellular Automata. Springer, Berlin (2010)
5. Toffoli, T., Margolus, N.: Cellular Automata Machine. MIT Press, USA (1987)

Comparative Study of Parallel Algorithms for Asynchronous Cellular Automata Simulation on Different Computer Architectures*

Konstantin Kalgin

Institute of Computational Mathematics and Mathematical Geophysics SB RAS
kalgin@ssd.sccc.ru

Abstract. Overview and experimental comparative study of parallel algorithms of asynchronous cellular automata simulation is presented. The algorithms are tested for the model of physicochemical process of surface $CO + O_2$ reaction over the supported Pd nanoparticles on different parallel computers. For testing we use shared memory computers, distributed memory computers (i.e. clusters), and graphical processing unit. Characterization of these algorithms in respect of methods of parallelism maintenance is given.

1 Introduction

Asynchronous cellular automata (ACA) are used for simulation of physical and chemical processes on molecular level, for example, to study oscillatory chemical surface reactions [1,2], absorption, sublimation and diffusion of atoms in the epitaxial growth processes [3]. Simulation of natural processes requires huge cellular space and millions of iterative steps for obtaining the real scene of the process. Therefore, it requires a lot of computing power. Unfortunately, ACA can not be parallelized so easily as synchronous cellular automata (SCA). As distinct to SCA, ACA functioning is a sequential application of transition rule to randomly selected cells. The cells are selected with equal probabilities and irrespective of the process history.

Parallelization of the ACA is performed by domain decomposition method: each process hosts its own domain of cells and stores the copies of boundary cells of neighboring processes. A parallel algorithm simulation should preserve the behavioral properties of ACA: **I**ndependence, **F**airness, **C**orrectness, and **E**fficiency. Independence means independent selection of cells during simulation. Fairness means that different cells are selected with equal probabilities. Correctness means deadlock-absence and coherence of boundary cell states and corresponding copies in different processes. Efficiency implies that T_k is less than T_1 for some k . Here T_k is the total time of parallel algorithm execution on

* Supported by (1)Presidium of Russian Academy of Sciences, Basic Research Program N 2 (2009), (2)Siberian Branch of Russian Academy of Sciences, SBRAS Interdisciplinary Project 32 (2009).

k processors, and T_1 is the total time of sequential algorithm execution on one processor.

There are several parallel algorithms of ACA simulation on computers with different architectures. In [4] an algorithm suitable for shared memory computers only is proposed. Parallel algorithms for distributed memory computers are presented in [5,6]. In addition, [7] and [8] describe a practical approach to parallel simulation of ACA. Where the given ACA is transformed into a synchronous one, called block-synchronous cellular automata (BSCA), that approximates its evolution and also provides easy parallelization.

This paper presents comparative study of the mentioned above parallel algorithms and their efficiency on computers with different architectures. Section 2 gives a formal definition of ACA. Section 3 gives main ideas of the mentioned algorithms and briefly characterizes them in respect of four properties given above. In section 4 the following parallel computer systems are overviewed: shared memory computers, distributed memory computers (i.e. clusters), and graphical processing unit (GPU) supporting CUDA [9]. These systems are used for testing implementations of the mentioned algorithms. Section 5 describes an ACA model of physicochemical process of surface $CO+O_2$ reaction over the supported Pd nanoparticles [2]. Also this section presents results of testing of the mentioned parallel algorithms implemented for this model on different parallel architectures. All tested combinations of computer architectures and parallel algorithm implementations are presented at Table 1. We do not present results for the algorithm given in [5] as far as additional costs of the parallel computations maintaining are too large for the model under consideration.

Table 1. Tested combinations of computer architectures and parallel algorithm implementations

	Shared memory	Distributed memory	Graphical processing unit
[4]	+	-	-
[6]	+	+	-
[7,8]	+	+	+

2 Asynchronous Cellular Automata

Asynchronous cellular automaton is specified by the following tuple:

$$ACA = \langle Z^d, A, \Theta \rangle,$$

where Z^d is a finite set of *cell coordinates*, and A is an *alphabet*, i.e. a finite set of cell states, and Θ is a transition rule.

Further we use a two dimensional rectangular space Z^2 :

$$Z^2 = \{(i, j) \mid 1 \leq i \leq N_x, 1 \leq j \leq N_y\}$$

A pair $(\mathbf{x}, a) \in Z^d \times A$ is called a *cell*, where $a \in A$ is a state of the cell and $\mathbf{x} \in Z^d$ are its coordinates. Set of cells $\Omega = \{(\mathbf{x}_i, a_i)\} \subset Z^d \times A$ is called a *cellular array* if there does not exist a pair of cells with equal coordinates and $\{\mathbf{x} \mid (\mathbf{x}, a) \in \Omega\} = Z^d$. Since between the cells in a cellular array and their coordinates there exists a one-to-one correspondence, we will further identify each cell with its coordinates.

The *transition rule* Θ is a probabilistic function:

$$\Theta : A^{|T|} \rightarrow A^{|T|}$$

Where the *template* T is a set of *naming functions* $\phi_i : Z^2 \rightarrow Z^2$, $T = \{\phi_1, \phi_2, \dots, \phi_{|T|}\}$. The template determines a *neighborhood* of a cell \mathbf{x} :

$$T(\mathbf{x}) = \{\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_{|T|}(\mathbf{x})\}.$$

Further we use the following templates:

$$T_{13}(\mathbf{x}) = \{\mathbf{x} + \mathbf{v}_0, \mathbf{x} + \mathbf{v}_1, \dots, \mathbf{x} + \mathbf{v}_{12}\}$$

$$T_5(\mathbf{x}) = \{\mathbf{x} + \mathbf{v}_0, \mathbf{x} + \mathbf{v}_1, \dots, \mathbf{x} + \mathbf{v}_4\}$$

$$T_1(\mathbf{x}) = \{\mathbf{x} + \mathbf{v}_0\}$$

$$V = \{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_{12}\} =$$

$$\{(0, 0), (0, 1), (1, 0), (0, -1), (-1, 0),$$

$$(1, 1), (1, -1), (-1, 1), (-1, -1), (0, 2), (2, 0), (0, -2), (-2, 0)\}$$

An *application of the transition rule to a cell* \mathbf{x} results in updating neighboring cells $T(\mathbf{x})$ with new states $\Theta(T(\mathbf{x}))$.

As usual the transition rule can be expressed as a *substitution* or as a *composition* of several transition rules. The most used rules of composition are *random execution* (R), *sequential execution* (S), and *randomly ordered sequential execution* (RS). These rules can be given by

$$\Theta_R = R(\Theta_1, p_1; \Theta_2, p_2; \dots, \Theta_n, p_n) \tag{1}$$

$$\Theta'_R = R(\Theta_1, \Theta_2, \dots, \Theta_n) \tag{2}$$

$$\Theta_S = S(\Theta_1, \Theta_2, \dots, \Theta_n) \tag{3}$$

$$\Theta_{RS} = RS(\Theta_1, \Theta_2, \dots, \Theta_n) \tag{4}$$

$$T_{\Theta_R} = T_{\Theta'_R} = T_{\Theta_S} = T_{\Theta_{RS}} = \bigcup_{i=0}^n T_{\Theta_i} \tag{5}$$

The result of Θ_R application to \mathbf{x} coincides with result of Θ_i application to \mathbf{x} with probability p_i . If probabilities p_i are omitted (2), then they are equal to $1/n$. The result of Θ_S application to \mathbf{x} coincides with sequential applications of $\Theta_1, \Theta_2, \dots, \Theta_n$ to \mathbf{x} . The result of Θ_{RS} application to \mathbf{x} coincides with sequential applications of randomly ordered $\Theta_1, \Theta_2, \dots, \Theta_n$ to \mathbf{x} .

Elementary transition rule can be written as a substitution in the following form:

$$\Theta_{sub} : \{a_1, a_2, \dots, a_n\} \xrightarrow{p} \{a'_1, a'_2, \dots, a'_n\}. \quad (6)$$

Application of Θ_{sub} to a cell \mathbf{x} results in replacing the states of the cells $T_{\Theta_{sub}}(\mathbf{x})$ with probability p . Here probability p and states a'_i can be functions of current states, $p = p(a_1, a_2, \dots, a_n)$, $a'_i = f_i(a_1, a_2, \dots, a_n)$.

An ACA simulation process is split into *iterations*. An iteration comprises $|Z^2| = N_x \cdot N_y$ transition rule applications to randomly chosen cells.

3 Parallel Algorithms

In papers [4,5] ACA is defined as a discrete event model that evolves in continuous time. Transition rule applications at different cells occur asynchronously at random times. These applications form a Poisson process for each cell. For different cells these Poisson processes are independent, and the application rate is the same for each cell. Parallelization of the ACA model is performed by domain decomposition: each process hosts its own domain of cells and the copies of boundary cells of neighboring processes. For correct simulation of Poisson process for each cell, every computing process controls its own local time. The *process' local time* is the next time of transition rule application to a newly selected cell from its domain. A process increments its own local time by an exponentially distributed pseudorandom number after each transition rule application.

In [4] an algorithm suitable for shared memory computers is proposed. Each process repeats the following steps while its own local time is less than predefined T_{max} : (1) selects a random cell from its domain, (2) waits for the situation when minimal local time of neighboring processes is greater than its own local time for the cells belonging to the domain boundary, (3) applies transition rule to the cell, and (4) increments its own local time according to Poisson distribution. Independence and Fairness of the algorithm are provided by independence and fairness of random cell selection from the domain and the way of local time incrementing. Correctness is provided by the synchronization based on domain's local time briefly described in step (2). Efficiency is provided by the property that boundary cells are selected infrequently in large domains.

In [5] a modified Time Warp algorithm is presented. Time Warp [10] is an optimistic parallel algorithm for simulation of any discrete event model on distributed memory computers. The main idea of the parallel Time Warp algorithm is as follows. In contrast to the previous algorithm, application of the transition rule and computing new local time are performed without waiting for neighboring processes (each process "hopes" that neighboring processes will not change boundary cell states). If a process changes a boundary cell state, then it sends a message to its neighbors called a *positive message*. When a process receives a positive message "from the future" (i.e., its own local time is less than that of the sender) it saves the message for the future processing. A situation when process receives a positive message "from the past" is called *causality error*. In

this situation a recovery mechanism should be initiated. Recovery from the prematurely executed steps results in two things to be rolled back: the cellular array and the messages sent to the other processes. Rolling back the cellular array is accomplished by periodically saving updated cell states and restoring boundary cell states valid for the rolled back local time. Rolling back previously sent positive messages is accomplished by sending *anti-messages*. If a process receives an anti-message that corresponds to an unprocessed positive message, then these two messages annihilate each other and the process proceeds. If there arrives an anti-message that corresponds to a positive message already processed, then the process has made an error and is to be also rolled back. A consequence of the recovery mechanism is that more anti-messages can be sent to other processes recursively. **I**ndependence and **F**airness of the algorithm are provided in the same way as in the previous algorithm. **C**orrectness is provided by recovery mechanism from causality errors (for detail see [5]). In few words, **E**fficiency is provided by optimistic behavior of processes. However, there are some costs for saving the history of message sendings and updating of boundary cell states. If the costs are large (with respect to effective work), then efficiency of the algorithm decreases. Also efficiency significantly depends on the following two parameters of the transition rule [5]: amount of work to be performed for the rule computing and average number of actually changed cell states after application of the rule.

In [6] an algorithm suitable for distributed memory computers is presented. ACA is defined as described in Section 2. Let us consider a sequence of randomly selected coordinates $X = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ for an iteration. According to the decomposition of the cellular array into domains d_1, d_2, \dots, d_p , we can divide the sequence X into $2p$ parts: for each domain d_k we take its internal I_k and boundary B_k subsequences. The subsequences I_k and B_k can be formed by use of uniform, exponential, and binomial pseudorandom number generators. The algorithm is based on the stochastic properties of I_k and B_k , and on planning the order of interactions between processes. The main idea of the parallel algorithm is as follows. Firstly, a process k (hosting a domain d_k) forms the pair (I_k, B_k) independently of other processes. Secondly, each process avoids unnecessary synchronizations because it is informed about neighbors' subsequences $B_{k'}$. It means that the process waits only for those new boundary cell states that will be actually used by it. Note that in the first [4] algorithm process has to wait in any case. **I**ndependence and **F**airness of the algorithm are provided by the method in which I_k and B_k are formed, for detail see [6]. **C**orrectness is provided by means of synchronization based on B_k . **E**fficiency is provided by avoiding of unnecessary synchronizations.

In [7,8] a practical analogue of ACA is described. This model is called Block-Synchronous Cellular Automata due to the ways of asynchronism reduction. Iteration of BSCA is a sequence of m stages. On each stage the transition rule is synchronously applied to all cells from a randomly selected set $S_i \subset \Omega$, where $\{S_1, S_2, \dots, S_m\}$ is a partitioning of cellular array Ω with the following properties:

$$\bigcup_{i=1}^m S_i = \Omega \tag{7}$$

$$\forall i \forall j : S_i \cap S_j = \emptyset \tag{8}$$

$$\forall i \forall j : |S_i| = |S_j| \tag{9}$$

$$\forall i \forall a \in S_i \forall b \in S_i : T(a) \cap T(b) = \emptyset \tag{10}$$

Such a reduction of asynchronism (at the expense of **I**ndependence) results in very simple parallel algorithm. In this algorithm processes have to synchronize each with other (send and receive new boundary cell states) only between stages. High **e**fficiency of the algorithm is provided by rare process synchronizations. **C**orrectness and **F**airness are provided by properties (10) and (7,9), respectively.

4 Computers' Architecture Overview

For testing the algorithms given above, we use computers with three types of architecture: multicores and multiprocessors with shared memory, a cluster with distributed memory and graphical processing unit (GPU). Parameters of computers with shared memory *Core-i7* and *SMP-8* are given at the Table 2.

Table 2. Multicores' and multiprocessors' parameters

	Processors	GHz	Total cores	Memory controller
Core-i7	1×Intel Core i7	2.6	4	integrated
SMP-8	2×Intel Xeon 5140	2.3	8	separate

Cluster *MVS-100k* consists of *SMP-8* nodes connected through Infiniband.

GPU *GTX-280* consists of a 240-core processor and 2Gb off-chip *global memory*. The cores are grouped in 8-core *multiprocessors*. Each multiprocessor has its own 16Kb on-chip *shared memory* and 32Kb *register file*. Note, that multiprocessor belongs to SIMD (Single Instruction Multiple Data) class in Flynn's taxonomy: at each clock all eight cores perform the same instruction but using different arguments.

Parallel program intended for running on GTX-280 consists of thousands of threads grouped in *blocks*. Each block consists of not more than 512 threads. One block of threads can be run on one multiprocessor only. But one multiprocessor can manage up to 8 blocks. Threads of the same block can communicate all to all through the shared memory of multiprocessor and synchronize. But threads from different blocks can not communicate and can not synchronize in the same way.

Unfortunately, one can not directly implement parallel algorithms of ACA simulation mentioned above with exception to BSCA. The reason is in SIMD architecture of multiprocessor and impossibility of synchronization of threads belonging to different blocks.

5 Simulation of Surface Reactions on Palladium

The model of oscillatory dynamics of the $CO + O_2$ reaction over the supported Pd nanoparticles is described in [2]. This model is a combination of the model for the $CO + O_2$ reaction over the Pd(110) single crystal [11] and the stochastic model for the imitating the supported nanoparticle with dynamically changing shape and surface morphology [12]. The model consists of the following processes: CO adsorption ($\Theta_1, \Theta_2, \Theta_6$), CO desorption ($\Theta_3, \Theta_4, \Theta_7$), O_2 adsorption (Θ_9^i), CO diffusion ($\Theta_{11}^i, \Theta_{12}^i, \Theta_{13}^i$), Pd's atoms diffusion (Θ_{14}^i), subsurface oxygen O_{ss} formation (Θ_5), $CO + O$ and $CO + O_{ss}$ reaction ($\Theta_8, \Theta_{10}^i, \Theta_{15}^i$). In terms of ACA this model can be described as follows.

State a of a cell \mathbf{x} is written as $[n, \alpha]$, where n is the number of Pd atoms, $n \in \{0, 1, 2, \dots\}$, and α is the state of the Pd surface, $\alpha \in \{\emptyset, CO, O, O_{ss}, CO.O_{ss}\}$.

$$A = \{0, 1, 2, \dots\} \times \{\emptyset, CO, O, O_{ss}, CO.O_{ss}\}$$

$$\Theta = R(\Theta^1, \Theta^2, \Theta^3, \Theta^4)$$

$$\Theta^i = S(R(\Theta_1, p_1; \dots \Theta_8, p_8; \Theta_9^i, p_9; \Theta_{10}^i, p_{10}; S(\Theta_{11}^i, \dots \Theta_{14}^i), p_{11}), \Theta_{15}^i)$$

$$\Theta_{15}^i = RS\left(\Theta_{15}^{1,0}, \dots, \Theta_{15}^{5,0}, \Theta_{15}^{0,1}, \dots, \Theta_{15}^{0,5}, \Theta_{15}^{1,i}, \dots, \Theta_{15}^{5,i}, \Theta_{15}^{i,1}, \dots, \Theta_{15}^{i,5}\right)$$

$$T_{\Theta}(\mathbf{x}) = T_{\Theta^i}(\mathbf{x}) = T_{13}(\mathbf{x})$$

$$T_{\Theta_j}(\mathbf{x}) = T_1(\mathbf{x})$$

$$T_{\Theta_j^i}(\mathbf{x}) = T_1(\mathbf{x}) \cup T_1(\mathbf{x} + \mathbf{v}_i), \quad j = 9, 10, 11, 12, 13$$

$$T_{\Theta_j^i}(\mathbf{x}) = T_5(\mathbf{x}) \cup T_5(\mathbf{x} + \mathbf{v}_i), \quad j = 14, 15$$

$$T_{\Theta_{15}^{k,m}}(\mathbf{x}) = T_1(\mathbf{x} + \mathbf{v}_k) \cup T_1(\mathbf{x} + \mathbf{v}_m)$$

$\Theta_1 : \{[n, \emptyset]\} \rightarrow \{[n, CO]\}$	$\Theta_9^i : \{[n, \emptyset], [n, \emptyset]\} \rightarrow \{[n, O], [n, O]\}$
$\Theta_2 : \{[0, \emptyset]\} \rightarrow \{[0, CO]\}$	$\Theta_{10}^i : \{[n, CO], [n, O_{ss}]\} \rightarrow \{[n, \emptyset], [n, \emptyset]\}$
$\Theta_3 : \{[n, CO]\} \rightarrow \{[n, \emptyset]\}$	$\Theta_{11}^i : \{[n, CO], [m, \emptyset]\} \rightarrow \{[n, \emptyset], [m, CO]\}$
$\Theta_4 : \{[0, CO]\} \rightarrow \{[0, \emptyset]\}$	$\Theta_{12}^i : \{[n, CO], [m, O_{ss}]\} \rightarrow \{[n, \emptyset], [m, CO.O_{ss}]\}$
$\Theta_5 : \{[n, O]\} \rightarrow \{[n, O_{ss}]\}$	$\Theta_{13}^i : \{[n, CO.O_{ss}], [m, O_{ss}]\} \rightarrow \{[n, O_{ss}], [m, CO.O_{ss}]\}$
$\Theta_6 : \{[n, O_{ss}]\} \rightarrow \{[n, CO.O_{ss}]\}$	
$\Theta_7 : \{[n, CO.O_{ss}]\} \rightarrow \{[n, O_{ss}]\}$	$\Theta_{14}^i : \{[n+1, \emptyset], a_1, \dots, a_4, [m, \emptyset], a_6, \dots, a_9\} \xrightarrow{p'} \{[n, \emptyset], a_1, \dots, a_4, [m+1, \emptyset], a_6, \dots, a_9\}$
$\Theta_8 : \{[n, CO.O_{ss}]\} \rightarrow \{[n, \emptyset]\}$	$\Theta_{15}^{k,j} : \{[n, CO], [n, O]\} \rightarrow \{[n, \emptyset], [n, \emptyset]\}$

Here for Θ_{14}^i the states a_i are $a_i = [n_i, \alpha_i]$, and the probability p' is the probability of actual Pd atom movement, which depends on changing of total energy of atoms connections ΔE , $p' = \exp(-\Delta E/kT)$. The probabilities p_i depend on rates of processes k_i , $p_i = k_i / \sum_{j=1}^{11} k_j$. For concrete values k_1, k_2, \dots, k_{11} and concrete energies of atoms connections see [2].

The algorithm [4] is implemented as a multithreaded program using POSIX Threads. Each thread controls its own local time and hosts a part of the cellular array (domain). The algorithm [6] is implemented using MPI (Message Passing Interface). For sending short messages containing new cell states MPI_Bsend (buffered mode) is used. The algorithm [7,8] is implemented using MPI and OpenMP. In each SMP-8 node one process with eight threads is executed. Using of OpenMP reduces the number of actually executed processes and therefore, also reduces communication costs. For GTX-280 the algorithm [7,8] is implemented using CUDA [9]. Each thread deals with the neighborhood of a particular cell. First, it loads states of the neighboring cells to the shared memory. Then the thread computes new states for the neighborhood. After that the thread puts new states back to the global memory.

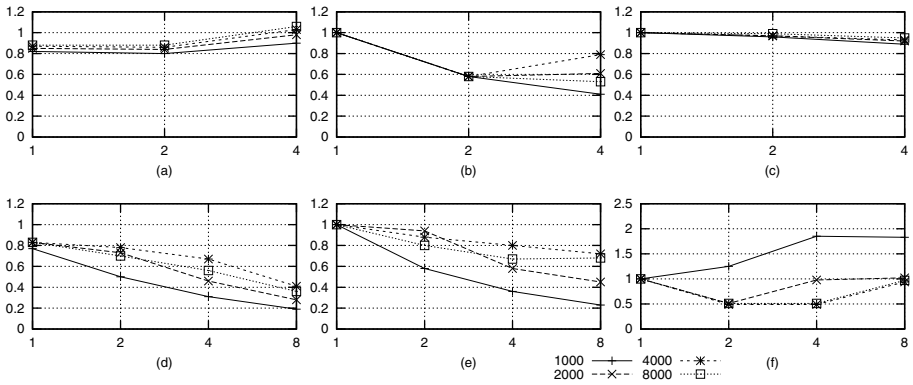


Fig. 1. Efficiency of the parallel algorithms ((a, d) for [4], (b, e) for [6], and (c, f) for [7,8]) for Core-i7 (a, b, c) and SMP-8 (d, e, f). Along x-axis are the numbers of used cores, along y-axis efficiency is shown.

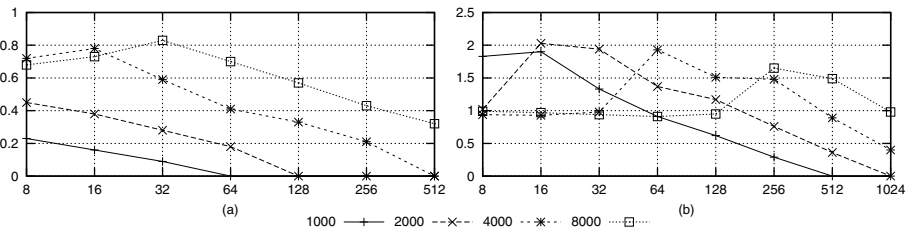


Fig. 2. Efficiency of the parallel algorithms ((a) for [6], and (b) for [7,8]) for MVS-100k. Along x-axis are the numbers of used cores, along y-axis efficiency is shown.

Results of testing on shared memory computers, cluster and GPU are presented at Fig. 1, Fig. 2, and Table 3, respectively. All tests are performed with several cellular array sizes (1000×1000 , 2000×2000 , 4000×4000 , 8000×8000). As usual, the efficiency of parallelization is $E_p = T_1/(pT_p)$.

Table 3. Acceleration of the [7,8] algorithm implemented on GPU in comparison with that on Core-i7

	1000	2000	4000	8000
GTX-280	25	31	34	35

6 Conclusion

Results of testing show that efficiency of the algorithm [4] is high on modern multicore computers (Core-i7) even for relatively small cellular arrays. The algorithm [6] delivers good efficiency only for large cellular arrays but can be run on cluster with several nodes. The reason is in additional costs of MPI sendings and receivings. Further improvement of the algorithm should be focused on multithread extensions to reduce the costs of communications. The algorithm [7,8] shows high efficiency for all sizes of cellular array and on all parallel architectures. The reason of such high performance is in reduction of asynchronism (at the expense of Independence). For some models it is shown [7,8] that such reduction does not affect the simulation process. Nevertheless, for each new model one has to make sure that the model allows such reduction.

Acknowledgment. I would like to thank Dr. O. L. Bandman and Dr. V. I. Elokhin for fruitful discussions and advise.

References

1. Danielak, R., Perera, A., Moreau, M., Frankowicz, M., Kapral, R.: Surface Structure and Catalytic CO Oxidation Oscillations, February 13 (1996), arXiv:chaodyn/9602015v1
2. Elokhin, V.I., Latkin, E.I., Matveev, A.V., Gorodetskii, V.V.: Application of Statistical Lattice Models to the Analysis of Oscillatory and Autowave Processes on the Reaction of Carbon Monoxide Oxidation over Platinum and Palladium Surfaces. *Kinetics and Catalysis* 44(5), 672–700 (2003)
3. Neizvestny, I.G., Shwartz, N.L., Yanovitskaya, Z.S., Zverev, A.V.: 3D-model of epitaxial growth on porous {111} and {100} Si surfaces. *Computer Physics Communications* 147, 272–275 (2002)
4. Lubachevsky, B.D.: Efficient parallel simulations of asynchronous cellular arrays. *Complex Systems* 1(6), 1099–1123 (1987)
5. Overeinder, B.J., Sloot, P.M.A.: Extensions to Time Warp Parallel Simulation for Spatial Decomposed Applications. *Proceedings of the Fourth United Kingdom Simulation Society Conference (UKSim 1999)*, pp. 67–73 (1999)
6. Kalgin, K.V.: Parallel simulation of asynchronous Cellular Automata evolution. *Bull. Comp. Center, Nov. Comp. Science* 27, 55–63 (2008)

7. Nedeia, S.V., Lukkien, J.J., Hilbers, P.A.J., Jansen, A.P.J.: Methods for Parallel Simulations of Surface Reactions, September 4 (2002), arXiv:physics/0209017v1
8. Bandman, O.L.: Parallel Simulation of Asynchronous Cellular Automata Evolution. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 41–47. Springer, Heidelberg (2006)
9. NVIDIA CUDA Programming Guide, http://www.nvidia.com/object/cuda_get.html (accessed May 30, 2010)
10. Jefferson, D.R.: Virtual time. ACM Trans. on Program. Lang. and Sys. 7(3), 404–425 (1985)
11. Latkin, E.I., Elokhin, V.I., Matveev, A.V., Gorodetskii, V.V.: J. Molec. Catal. A. Chemical 158, 161–166 (2000)
12. Kovalyov, E.V., Elokhin, V.I., Myshlyavtsev, A.V.: J. Comput. Chem. 29, 79–86 (2008)

Coxeter Groups and Asynchronous Cellular Automata

Matthew Macauley¹ and Henning S. Mortveit²

¹ Department of Mathematical Sciences, Clemson University

² Department of Mathematics and NDSSL/VBI, Virginia Tech.

Abstract. The dynamics group of an asynchronous cellular automaton (ACA) relates properties of its long term dynamics to the structure of Coxeter groups. The key mathematical feature connecting these diverse fields is involutions. Group-theoretic results in the latter domain may lead to insight about the dynamics in the former, and vice-versa. In this article, we highlight some central themes and common structures, and discuss novel approaches to some open and open-ended problems. We introduce the state automaton of an ACA, and show how the root automaton of a Coxeter group is essentially part of the state automaton of a related ACA.

Keywords: Asynchronous cellular automaton, Coxeter group, dynamics group, sequential dynamical system.

1 Introduction

An asynchronous cellular automaton (ACA) is defined in the same manner as a classical cellular automaton (CA) in all aspects except the evaluation mechanism. As the name suggests, the maps associated to the vertices (or nodes) are applied synchronously for a CA, and asynchronously for an ACA. In general, there are many ways that one can apply maps asynchronously. For example, one may select a vertex at random according to some probability distribution, apply the corresponding map, and repeat this procedure. Alternatively, one may select a fixed permutation over the vertices and apply the maps in the sequence specified by this permutation. This permutation evaluation process would correspond to increasing the time by one unit, and would be applied repeatedly to generate the system dynamics. An important aspect of having a fixed permutation update sequence is that one obtains a dynamical system. This is not necessarily the case in the more general situation, such as when the individual states are updated at random.

The analysis of CAs and ACAs does not have the support that the study of ODEs has from established fields such as analysis and differential geometry. As such, a key goal of CA/ACA research is to make connections to existing mathematical theory. We will consider the class of π -independent ACAs – those whose periodic points (as a set) are independent of the permutation update sequence. While this may seem to be a rather exotic property, we have shown that

roughly 40% of the elementary CA rules give rise to π -independent ACAs [8]. Given a π -independent ACA, one can define its *dynamics group*. This permutation group on the set of periodic points is a quotient of a Coxeter group, and it captures the possible long-term dynamics that one can generate by suitable choices of update sequence. Its structure can answer questions about the existence and non-existence of periodic orbits of given sizes.

In this paper, we will revisit the notions of Coxeter systems and sequential dynamical systems (SDSs). An SDS is a generalization of an ACA (assuming a fixed update sequence) where the underlying graph is arbitrary, and is not limited to being a regular lattice or circle (i.e., a one-dimensional torus). We will show how the word problem for Coxeter groups is related to functional equivalence of SDS maps. This forms the basis for our next result, on how conjugation of Coxeter elements corresponds to cycle equivalence of SDS maps, and additionally, how this extends from conjugacy classes to spectral classes. After defining dynamics groups and showing how they arise as quotients of Coxeter groups, we show how key features of mathematical objects in both the fields of SDSs and Coxeter groups are encoded by finite (or infinite) state automata. We illustrate this by explicit examples, and then close with a table summarizing these connections.

2 Background

A *Coxeter system* is a pair (W, S) consisting of a group W generated by a set $S = \{s_1, \dots, s_n\}$ of involutions given by the following presentation

$$W = \langle s_1, \dots, s_n \mid s_i^2 = 1, (s_i s_j)^{m(s_i, s_j)} = 1 \rangle,$$

where $m(s_i, s_j) \geq 2$ for $i \neq j$. Let S^* be the free monoid over S , and for each integer $m \geq 0$ and distinct generators $s, t \in S$, define

$$\langle s, t \rangle_m = \underbrace{stst \cdots}_m \in S^*.$$

The relation $\langle s, t \rangle_{m(s,t)} = \langle t, s \rangle_{m(s,t)}$ is called a *braid relation*, and is additionally called a *short braid relation* if $m(s, t) = 2$. Note that s and t commute if and only if $m(s, t) = 2$. A Coxeter system can be described uniquely by its *Coxeter graph* Γ , which has vertex set $V = \{1, \dots, n\}$ and an edge $\{i, j\}$ for each non-commuting pair of generators $\{s_i, s_j\}$, with edge label $m(s_i, s_j)$.

Switching to ACAs and SDSs, let Γ be an undirected graph (called the *base graph* or *dependency graph*) with vertex set $V = \{1, \dots, n\}$. We equip each vertex i with a state $x_i \in K$ where K is a set called the *state space*, and a *vertex function* f_i that maps (or updates) $x_i(t)$ to $x_i(t + 1)$ based on the states of its neighbors (itself included). Unless explicitly stated otherwise, we will assume that $K = \mathbb{F}_2 = \{0, 1\}$, which is the most commonly used state space in cellular automata research. If the vertex functions are applied asynchronously, it is convenient to encode f_i as a Γ -local function $F_i: K^n \rightarrow K^n$ defined by

$$F_i(x_1, \dots, x_n) = (x_1, \dots, x_{i-1}, f_i(x_1, \dots, x_n), x_{i+1}, \dots, x_n).$$

If f_i does not depend on all n states, it may be convenient to omit the fictitious variables. Given a sequence of local functions and a word $w = w_1w_2 \dots w_m \in V^*$ called the *update sequence*, the SDS map F_w is the composition of the local functions in the order prescribed by w , i.e.,

$$F_w : K^n \longrightarrow K^n, \quad F_w = F_{w_m} \circ F_{w_{m-1}} \circ \dots \circ F_{w_2} \circ F_{w_1}.$$

SDSs represent a generalization of ACAs, which are usually defined over a regular grid, such as \mathbb{Z} or \mathbb{Z}_n . The following example illustrates some SDS concepts – see [14] for a more complete treatment.

Example 1. We take $\Gamma = \text{Circ}_4$ as base graph (see Figure 1) and use $K = \{0, 1\}$ as the state space. Also, we take all vertex functions to be Boolean nor-functions given by $\text{nor} : K^3 \rightarrow K$ where $\text{nor}(x, y, z)$ equals 1 if $x = y = z = 0$ and 0 otherwise. In this case we have, for example, $F_1(x_1, x_2, x_3, x_4) = (\text{nor}(x_4, x_1, x_2), x_2, x_3, x_4)$. Using the update sequence $\pi = 1234$, we obtain

$$\begin{aligned} F_1(0, 0, 0, 0) &= (1, 0, 0, 0) \\ F_2 \circ F_1(0, 0, 0, 0) &= (1, 0, 0, 0) \\ F_3 \circ F_2 \circ F_1(0, 0, 0, 0) &= (1, 0, 1, 0) \\ F_4 \circ F_3 \circ F_2 \circ F_1(0, 0, 0, 0) &= (1, 0, 1, 0), \end{aligned}$$

and thus $F_\pi(0, 0, 0, 0) = (1, 0, 1, 0)$. The phase space of the map F_π is the directed graph containing all global state transitions and is shown in Figure 1.

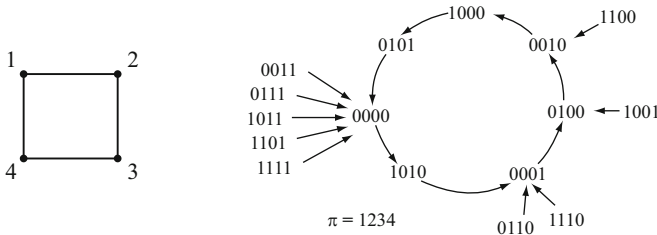


Fig. 1. The base graph $\Gamma = \text{Circ}_4$ and the phase space of F_π from Example 1

3 The Word Problem

A fundamental question given any finitely presented group $\langle S \mid R \rangle$, is when do two words

$$w = w_1w_2 \dots w_m, \quad \text{and} \quad w' = w'_1w'_2 \dots w'_k$$

in S^* yield the same group element? This is the *word problem*, and it is in general undecidable. However, there are many classes of groups for which the word problem is solvable. A classic result in Coxeter groups, known as *Matsumoto’s theorem* [5, Theorem 1.2.2], says that any two reduced expressions for the same

element differ by braid relations. Matsumoto’s theorem provides an algorithmic solution to the word problem for Coxeter groups.

There is an analog of the word problem for SDSs. Specifically, given two update sequences $w, w' \in V^*$, when are the corresponding SDS maps

$$F_w = F_{w_m} \circ F_{w_{m-1}} \circ \dots \circ F_{w_2} \circ F_{w_1}, \quad F_{w'} = F_{w'_k} \circ F_{w'_{k-1}} \circ \dots \circ F_{w'_2} \circ F_{w'_1}$$

equal as functions, or equivalently, when do they have identical phase spaces? This is clearly solvable because there are only finitely many functions $\mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$. However, it would be desirable to solve this problem algorithmically for general SDSs, without resorting to checking the image of all 2^n global states.

4 Equivalences on Dynamics and Acyclic Orientations

In this section, we show how topological conjugation of SDS maps corresponds to conjugation of elements in a Coxeter group, and how this connection leads to a coarser equivalence relation when the graph Γ has non-trivial symmetries. *Acyclic orientations* are mathematically convenient to capture several types of equivalences on permutation SDS maps, as well as on Coxeter elements in Coxeter groups. A *Coxeter element* is the product of the generators of S in some order. Every Coxeter element defines a partial ordering on S , which we can represent by an acyclic orientation of Γ . Specifically, for a Coxeter element c , define the orientation (Γ, c) so that edge $\{i, j\}$ is oriented (i, j) if s_i appears before s_j in c . It is easy to show that this is well-defined, and that it defines a bijection between the set $\text{Acyc}(\Gamma)$ of acyclic orientations of Γ and the set $C(W)$ of Coxeter elements of W .

Next, consider conjugating a Coxeter element $c = s_{x_1} \dots s_{x_n}$ by the initial letter $s = s_{x_1}$, which results in a cyclic shift of the word:

$$scs = s_{x_1}(s_{x_1}s_{x_2} \dots s_{x_n})s_{x_1} = s_{x_2}s_{x_3} \dots s_{x_n}s_{x_1}.$$

The corresponding acyclic orientations (Γ, c) and (Γ, scs) differ by converting the source vertex of (Γ, c) into a sink. This source-to-sink conversion generates an equivalence relation \sim_κ on $\text{Acyc}(\Gamma)$, and it was recently proven (see [4]) that $(\Gamma, c) \sim_\kappa (\Gamma, c')$ if and only if c and c' are conjugate. (Note that the “if” direction is obvious; the “only if” direction is difficult).

Turning to SDSs, let $S_n \subset V^*$ be the set of words where each vertex appears precisely once, which we may identify with the permutations of V . Each permutation $\pi \in S_n$ defines a partial ordering on V , and there is a natural map from $\text{Acyc}(\Gamma)$ to the set of permutation SDS maps (π is mapped to F_π). Two finite dynamical systems $\phi, \psi: K^n \rightarrow K^n$ are said to be *cycle equivalent* if for some bijection $h: K^n \rightarrow K^n$ we have $\psi|_{\text{Per}(\psi)} \circ h = h \circ \phi|_{\text{Per}(\phi)}$, where $\text{Per}(\phi)$ denotes the set of periodic states of ϕ . The following result provides the connection between κ -equivalence of acyclic orientations and cycle equivalence of permutation SDS maps.

Theorem 1 ([12]). *If $(\Gamma, \pi) \sim_{\kappa} (\Gamma, \sigma)$, then F_{π} and F_{σ} are cycle equivalent.*

If the automorphism group $\text{Aut}(\Gamma)$ is non-trivial, we can say even more. The group $\text{Aut}(\Gamma)$ acts on $\text{Aut}(\Gamma)/\sim_{\kappa}$ by $\gamma \cdot [(\Gamma, \pi)] = [(\Gamma, \gamma\pi)]$, which gives rise to the equivalence relation $\sim_{\bar{\kappa}}$ on $\text{Aut}(\Gamma)/\sim_{\kappa}$. This coarser equivalence relation also has an interpretation in the settings of both Coxeter groups and SDSs.

If (W, S) is a Coxeter system with $|S| = n$, let \mathcal{V} be an n -dimensional real vector space with basis $\{\alpha_1, \dots, \alpha_n\}$. Put a symmetric bilinear form B on \mathcal{V} , defined by $B(\alpha_i, \alpha_j) = -\cos(\pi/m(s_i, s_j))$. The group W acts on \mathcal{V} by

$$s_i: \mathbf{v} \mapsto \mathbf{v} - 2B(\mathbf{v}, \alpha_i)\alpha_i, \tag{1}$$

and the set of elements $\Phi = \{w\alpha_i \mid w \in W, i = 1, \dots, n\}$ are called *roots*. This action is faithful and preserves the bilinear form B . Geometrically, the root $s_i\mathbf{v}$ is the reflection of \mathbf{v} across the hyperplane α_i^{\perp} , and so there is a representation $\rho: W \rightarrow \text{GL}(\mathcal{V})$, defined on the generators by

$$\rho: s_i \longmapsto (\mathbf{v} \xrightarrow{F_i} \mathbf{v} - 2B(\mathbf{v}, \alpha_i)\alpha_i), \tag{2}$$

called the *standard geometric representation* of W (see [17]). This allows us to view elements in W as matrices, and hence we can speak of the characteristic polynomial of any given $w \in W$.

Now, if (Γ, c) and (Γ, c') differ by some $\gamma \in \text{Aut}(\Gamma)$, then $\rho(c)$ and $\rho(c')$ are similar as linear transformations. Specifically, they are conjugate in $\text{GL}(\mathcal{V})$ by the permutation matrix P_{γ} of γ . In this case, we say that c and c' have the same spectral class, because $\rho(c)$ and $\rho(c')$ have the same multiset of eigenvalues. Clearly, this is a weaker condition than conjugacy, and so all Coxeter elements in the same $\bar{\kappa}$ -equivalence class have the same spectral class.

Similarly, in the context of SDSs, if $(\Gamma, \pi) \sim_{\bar{\kappa}} (\Gamma, \sigma)$, then the SDS maps F_{π} and F_{σ} are cycle equivalent, due to the following argument. If $\gamma \in \text{Aut}(\Gamma)$, then the permutations π and $\gamma\pi$ give topologically conjugate SDS maps, F_{π} and $F_{\gamma\pi}$. Strictly speaking, this requires the maps f_i to be $\text{Aut}(\Gamma)$ -invariant (see [12]), a condition which is frequently satisfied in practice, such as when all vertices of the same degree share the same symmetric function (e.g., logical AND, OR, Majority, Parity, threshold functions, etc.). Since topologically conjugate maps are cycle equivalent, our statement follows.

It is worth mentioning the role of the Tutte polynomial here [15]. The Tutte polynomial of a graph Γ is a 2-variable polynomial $T_{\Gamma}(x, y)$ that satisfies a recurrence under edge deletion and contraction, and plays a central role in graph theory. Many graph counting problems are simply the evaluation of the Tutte polynomial at some $(x_0, y_0) \in \mathbb{Z} \times \mathbb{Z}$. For example, $|\text{Acyc}(\Gamma)| = T_{\Gamma}(2, 0)$ and $|\text{Acyc}(\Gamma)/\sim_{\kappa}| = T_{\Gamma}(1, 0)$. Thus, $T_{\Gamma}(2, 0)$ counts the number of Coxeter elements in the Coxeter group with Coxeter graph Γ , and it bounds the number of permutation SDS maps in an SDS with dependency graph Γ . This bound is known to be sharp for certain classes of functions [14]. Similarly, $T_{\Gamma}(1, 0)$ counts the number of conjugacy classes of Coxeter elements (see [4,10]), and it bounds the number of cycle equivalence classes of SDS maps (see [12]).

5 Groups

A sequence $F = (F_1, \dots, F_n)$ of local functions is π -independent if $\text{Per}(F_\pi) = \text{Per}(F_\sigma)$ for all $\pi, \sigma \in S_n$. Note that this is an equality of sets; we do not assume anything about the organization of the respective periodic points into periodic orbits. In this case, each F_i permutes the periodic points, and these permutations generate the *dynamics group* of F , denoted $\mathcal{DG}(F)$. Let F_i^* denote the restriction of F_i to $\text{Per}(F_\pi)$. Because F_i only changes the i^{th} coordinate of a state, and since we assume that $K = \mathbb{F}_2$, $F_i^* \circ F_i^*$ is the identity function on $\text{Per}(F_\pi)$. If we define $m_{ij} := |F_i^* \circ F_j^*|$, then there is a surjection

$$\langle s_1, \dots, s_n \mid s_i^2 = 1, (s_i s_j)^{m_{ij}} = 1 \rangle \longrightarrow \mathcal{DG}(F), \tag{3}$$

showing that dynamics groups are quotients of Coxeter groups. The particular homomorphism is determined by adding relations to the presentation of the Coxeter group, and these relations arise because the state space is \mathbb{F}_2 . Thus, dynamics groups are in a sense “reflection groups over \mathbb{F}_2 .” An open-ended research problem is to give an efficient presentation of the dynamics groups of an SDS based on the functions, i.e., to determine these extra relations.

When the base graph Γ of an SDS is the circular graph \mathbb{Z}_n , and the local functions are all identical, the resulting SDS is an *elementary ACA*. Each local function F_i takes $\{x_{i-1}, x_i, x_{i+1}\}$ as input, and is completely described by the following rule table

$x_{i-1}x_i x_{i+1}$	111	110	101	100	011	010	001	000
$f_i(x_{i-1}, x_i, x_{i+1})$	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

Clearly, there are $2^{2^3} = 256$ such choices of functions, which can be indexed by $k = \sum a_i 2^i \in \{0, \dots, 255\}$. The corresponding sequence of local functions is denoted ECA_k . In [8], it was shown that ECA_k is π -independent for precisely 104 values of k . Moreover, this holds for all $n > 3$. The dynamics groups of these 104 rules were classified in [11]. Among some of the interesting groups were $\mathcal{DG}(\text{ECA}_{60}) = \text{SL}_n(\mathbb{F}_2)$ and $\mathcal{DG}(\text{ECA}_k) = \mathbb{Z}_2^n$ for $k \in \{28, 29, 51\}$. Moreover, other dynamics groups were found computationally to be either the symmetric or alternating groups, with the size depending on the n^{th} Fibonacci or Lucas number, leading to a few conjectures.

6 The Root Automaton

The dynamics of all possible SDSs given a sequence of Γ -local functions $F = (F_1, \dots, F_n)$ can be encoded by the *state automaton* of the sequence. This is a directed graph Φ with vertex set K^n – the set of global system states, and directed edges $(x, F_i(x))$ for each $x \in K^n$ and each $i \in V$. Label such an edge with the index i corresponding to its vertex function; see Figure 2 for an example.

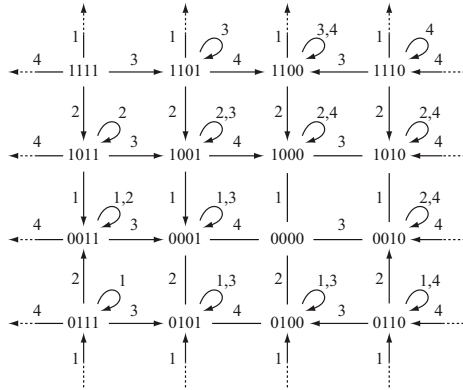


Fig. 2. The state automaton Φ for the SDS in Example 1. Horizontal/vertical dashed lines and arrows indicate horizontal/vertical wrap-around, and arrowheads are omitted from the bidirectional edges for clarity.

The image of a state $x \in K^n$ under an SDS map F_π , where $\pi = \pi_1\pi_2 \cdots \pi_n$, is represented on the state automaton by a path in Φ . Specifically, start at vertex x and traverse the path

$$x, F_{\pi_1}(x), F_{\pi_2}F_{\pi_1}(x), \dots, F_{\pi_n} \cdots F_{\pi_2}F_{\pi_1}(x) = F_\pi(x).$$

The phase space of F_π can be easily derived from the state automaton – it is the graph with vertex set K^n and an edge (x, y) for every directed path from a state x to y that traverses a path of edges labeled $\pi_1, \pi_2, \dots, \pi_n$. Note that if F is π -independent, then $\mathcal{DG}(F)$ acts on $\text{Per}(F)$. In this case, all of the edges within $\text{Per}(F)$ are bidirectional, and so we may view them as undirected.

This is the SDS analog of the action of W on $\Phi \subset \mathcal{V}$, as described in (11). Since \mathcal{V} is any n -dimensional vector space, we can identify it with \mathbb{R}^n , and assume that the basis elements are $\alpha_i = e_i$, the standard unit normal vectors. This associates roots with vectors in \mathbb{R}^n , and we partially order Φ by \leq componentwise ($z \preceq z'$ iff $z_i \leq z'_i$ for each i) to get the *root poset*. It is well-known that for every root, all non-zero entries have the same sign, thus we have a notion of positive and negative roots, and the root poset has a positive side Φ^+ and a negative side, Φ^- , with $\Phi = \Phi^+ \cup \Phi^-$. The image of s_i under the geometric representation from (2) is a linear map $F_i: \mathbb{R}^n \rightarrow \mathbb{R}^n$, where

$$F_i: (z_1, \dots, z_n) \mapsto (z_1, \dots, z_{i-1}, z_i + \sum_{j=1}^n 2 \cos(\pi/m_{i,j})z_j, z_{i+1}, \dots, z_n). \quad (4)$$

To summarize, F_i changes the i^{th} entry of a vector by flipping its sign and then adding each neighboring state z_j weighted by $2 \cos(\pi/m_{ij})$.

In 1993, Brink and Howlett proved that Coxeter groups are automatic [2], and soon after, H. Eriksson developed the *root automaton* [3]. The root automaton has vertex set Φ and edge set $\{(z, s_i z) \mid z \in \Phi, s_i \in S\}$. For convenience, label

each edge $(z, s_i z)$ with the corresponding generator s_i . It is clear that upon disregarding loops and edge orientations (all edges are bidirectional anyways), we are left with the Hasse diagram of the root poset. We represent a word $w = s_{x_1} s_{x_2} \cdots s_{x_m}$ in the root automaton by starting at the unit vector $e_{x_1} \in \Phi^+$ and traversing the edges labeled $s_{x_2}, s_{x_3}, \dots, s_{x_m}$ in sequence. Denote the root reached in the root poset upon performing these steps by $r(W, w)$. The sequence

$$e_{x_1} = r(W, s_{x_1}), \quad r(W, s_{x_1} s_{x_2}), \quad \dots, \quad r(W, s_{x_1} s_{x_2} \cdots s_{x_m}) = r(W, w),$$

is called the *root sequence* of w . If $r(W, s_{x_1} s_{x_2} \cdots s_{x_i})$ is the first negative root in the root sequence for w , then a shorter expression for w can be obtained by removing s_{x_1} and s_{x_i} . By the *exchange property* of Coxeter groups (see [17]), every word $w \in S^*$ can be made into a reduced expression by iteratively removing pairs of letters in this manner. Thus, the root automaton can algorithmically detect reduced words.

We conclude with an example that illustrates these concepts, and shows how the root automaton of a Coxeter group is essentially a connected component of the state automaton of an sequential dynamical system with state space $K = \mathbb{R}$.

Example 2. Let $W = H_4$, which has Coxeter graph as shown in Figure 3, and presentation (using a, b, c, d instead of s_1, s_2, s_3, s_4):

$$H_4 = \langle a, b, c, d \mid a^2, b^2, c^2, d^2, (ab)^5, (bc)^3, (cd)^3, (ac)^2, (ad)^2, (bd)^2 \rangle.$$

It is well-known (see [7]) that H_4 is a finite group of order 14400, and is the isometry group of the 120-cell and its dual, the 600-cell, two of the six regular 4-polytopes. Thus, the root poset Φ consists of 14400 roots. A portion of the root automaton is shown in Figure 4. Recall that the root automaton is built on top of the root poset – stripping away the self-loops and edge labels leaves the Hasse diagram of Φ . The dotted-line in Figure 4 shows the boundary between the positive roots Φ^+ and negative roots Φ^- . The non-loop edges of the root automaton are all bidirectional – arrowheads are omitted for clarity.

Consider the word $w = abdcabacba \in H_4$. Starting at $e_a = (1, 0, 0, 0)$ (see Figure 4), and traversing the edges labeled $b, d, c, a, b, a, c, b, c, a$ in sequence, we see that the first negative root in the root sequence of w is $r(W, abdcabacbc)$. Therefore, removing the first instance of a and the last instance of c from w results in $bdcabacba$, a shorter expression for w . It is easily checked that no matter where we begin in $bdcabacba$, the corresponding path in the root automaton consists of only positive roots. Therefore, $bdcabacba$ is a reduced word in H_4 .

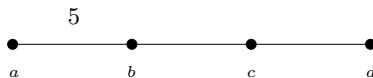


Fig. 3. The Coxeter graph Γ of the group $W = H_4$. As is customary, edge labels of 3 are suppressed.

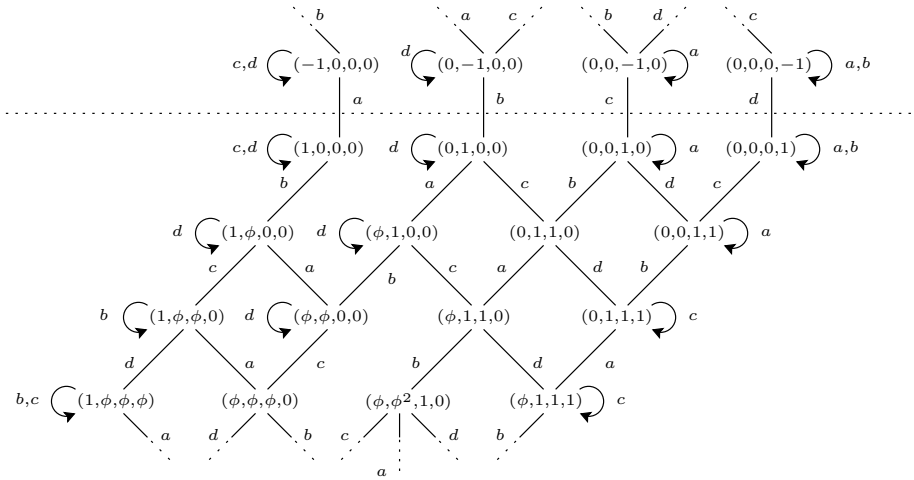


Fig. 4. Part of the root automaton of the group $W = H_4$. Here, $\phi = 2 \cos(\pi/5)$, the golden ratio. The dotted line separates the positive roots from the negative roots.

7 Summary

This paper presented a collection of results connecting properties of Coxeter groups and properties of the dynamics of ACAs/SDSs (see Table 1). These newly established connections provide possible avenues for ACA research. In a larger setting, we hope that our example linking properties of asynchronous, finite dynamical systems and group theory can provide inspiration for other approaches seeking to better understand the dynamics of ACAs through the use of existing mathematical structures and theory.

Table 1. Summary of the connections between Coxeter groups and SDSs

	Coxeter groups	Sequential dynamical systems
Graph $\Gamma \longleftrightarrow$	Coxeter graph	Dependency graph
Acyc(Γ) \longleftrightarrow	Coxeter elements $w = s_{\pi(1)} s_{\pi(2)} \cdots s_{\pi(n)}$	Permutation SDS maps $F_\pi = F_{\pi(n)} \circ \cdots \circ F_{\pi(2)} \circ F_{\pi(1)}$.
κ -equiv. \longleftrightarrow	Conjugacy classes of Coxeter elements	Cycle-equivalence classes of SDS maps
$\bar{\kappa}$ -equiv. \longleftrightarrow	Spectral classes of Coxeter elements	Cycle-equivalence classes of SDS maps (coarser)
$\Phi \longleftrightarrow$	Root poset / automaton	State automaton

Acknowledgments. The authors thank the Network Dynamics and Simulation Science Laboratory at the Virginia Bioinformatics Institute of Virginia Tech and Ilya Shmulevich's research group at the Institute for Systems Biology for their support.

References

1. Björner, A., Brenti, F.: *Combinatorics of Coxeter Groups*. Springer, Heidelberg (2005)
2. Brink, B., Howlett, R.B.: A finiteness property and an automatic structure for Coxeter groups. *Math. Ann.* 296, 179–190 (1993)
3. Eriksson, H.: *Computational and combinatorial aspects of Coxeter groups*. PhD thesis, KTH Stockholm (1994)
4. Eriksson, H., Eriksson, K.: Conjugacy of Coxeter elements. *Elect. J. Comb.* 16, #R4 (2009)
5. Geck, M., Pfeiffer, G.: *Characters of finite Coxeter groups and Iwahori-Hecke algebras*. Oxford Science Press (2000)
6. Hansson, A.Å., Mortveit, H.S., Reidys, C.M.: On asynchronous cellular automata. *Adv. Comp. Sys.* 8, 521–538 (2005)
7. Humphreys, J.E.: *Reflection Groups and Coxeter Groups*. Cambridge University Press, Cambridge (1990)
8. Macauley, M., McCammond, J., Mortveit, H.S.: Order independence in asynchronous cellular automata. *J. Cell. Autom.* 3, 37–56 (2008)
9. Macauley, M., McCammond, J., Mortveit, H.S.: Dynamics groups of asynchronous cellular automata. *J. Algebraic Combin.* (2010) (in press)
10. Macauley, M., Mortveit, H.S.: On enumeration of conjugacy classes of Coxeter elements. *Proc. Amer. Math. Soc.* 136, 4157–4165 (2008)
11. Macauley, M., Mortveit, H.S.: Posets from admissible Coxeter sequences (2010) (submitted)
12. Macauley, M., Mortveit, H.S.: Cycle equivalence of graph dynamical systems. *Nonlinearity* 22, 421–436 (2009)
13. Macauley, M., Mortveit, H.S.: Update sequence stability in graph dynamical systems. *Discrete Cont. Dyn. Sys. Ser. S* (2010) (in press)
14. Mortveit, H.S., Reidys, C.M.: *An Introduction to Sequential Dynamical Systems*. Springer, New York (2007)
15. Tutte, W.T.: A contribution to the theory of chromatic polynomials. *Canad. J. Math.* 6, 80–91 (1954)

Some Formal Properties of Asynchronous Cellular Automata

Luca Manzoni

Università degli Studi di Milano-Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione,
Viale Sarca 336, 20126 Milano, Italy
luca.manzoni@disco.unimib.it

Abstract. We study the dynamical behaviour of asynchronous cellular automata by considering some formal properties of classical cellular automata and adapting them to the asynchronous case.

1 Introduction

Cellular automata (CA) are a simple formal model for complex systems that is used in many scientific fields [21,9,10,3,29,25]. Synchronicity is one of the main features of this model. Indeed, a CA is made of identical finite automata which at the same (here comes synchronicity) time update their own state by a local rule on the basis of their current state and those of a fixed set of neighbours. For recent results on CA dynamics and an up-to-date bibliography see for instance [31,8,28,20,19,14,13,2,11,17,6,16,12,15,11,7].

Although CA have found success in modelling various phenomena, synchronicity may restrict their application on some real problems. In fact, a synchronous behaviour is a rare event in a real system. This pushed researchers to introduce asynchronism in the model. In the last 20 years, many empirical analyses [4,5,22,32,36,35] have been carried out. They highlighted that the behaviour of a CA considerably changes when relaxing the synchronicity constraint. This fact motivated subsequent theoretical studies on asynchronous CA. However, the few formal analyses of asynchronicity concern either examples [26,27,34] or peculiar classes [23,24] of probabilistic cellular automata.

In this paper we study a more general setting relaxing the synchronicity constraint. There are many possibilities of dealing with the asynchronicity in CA. Most of them involve probabilistic CA, i.e., the ones in which the updating of a cell happens with a certain probability p . In this work we consider a fully asynchronous updating: at each time step the local rule is applied only at one cell (a situation described by $p = 0$ in [35]). Indeed, we are interested in studying systems in which the synchronicity assumption of classical CA is completely overturned: no component is synchronized with any another one.

In classical CA the behaviour of the system is studied by some formal properties which give important information either on the CA global map (e.g., injectivity, surjectivity) or the CA dynamics (e.g., transitivity, sensitivity to initial conditions). These notions cannot be directly used for studying asynchronous CA. In this paper we suitably adapt them to the asynchronous case and we study them.

2 Basic Notions

In this section we briefly recall standard definitions about CA as discrete time dynamical systems. For all $i, j \in \mathbb{Z}$ with $i \leq j$ (resp., $i < j$), let $[i, j] = \{i, i + 1, \dots, j\}$ (resp., $[i, j) = \{i, i + 1, \dots, j - 1\}$). Let \mathbb{N}_+ be the set of positive integers.

Let A be an alphabet. A *configuration* is a function from \mathbb{Z} to A . The *configuration set* $A^{\mathbb{Z}}$ is usually equipped with the metric d defined as follows

$$\forall x, x' \in A^{\mathbb{Z}}, d(x, x') = 2^{-n}, \text{ where } n = \min \{i \geq 0 : x_i \neq x'_i \text{ or } x_{-i} \neq x'_{-i}\} .$$

If A is finite, $A^{\mathbb{Z}}$ is a compact, totally disconnected and perfect topological space (i.e. it is a Cantor space). For any pair $i, j \in \mathbb{Z}$, with $i \leq j$, and any configuration $x \in A^{\mathbb{Z}}$ we denote by $x_{[i, j]}$ the word $x_i \dots x_j \in A^{j-i+1}$.

A *1D CA* is a structure $\langle A, r, f \rangle$, where A is the *set of states* or *alphabet*, $r \in \mathbb{N}$ is the *radius* and $f : A^{2r+1} \rightarrow A$ is the *local rule* of the automaton. The local rule f induces a *global rule* $F : A^{\mathbb{Z}} \rightarrow A^{\mathbb{Z}}$ defined as follows,

$$\forall x \in A^{\mathbb{Z}}, \forall i \in \mathbb{Z}, F(x)_i = f(x_{i-r}, \dots, x_{i+r}) .$$

Note that F is a uniformly continuous map w.r.t. the metric d . For any CA, the structure $\langle A^{\mathbb{Z}}, F \rangle$ is a (discrete time) dynamical system. From now on, for the sake of simplicity, we identify a CA with the dynamical system induced by itself or even with its global rule F . A rule $f : A^{2r+1} \rightarrow A$ is *rightmost* (resp., *leftmost*) (resp., *center*) *permutive* iff $\forall u \in A^{2r}, \forall b \in A, \exists! c \in A$ such that $f(uc) = b$ (resp., $f(cu) = b$) (resp., $f(u_{[1, r]}cu_{[r+1, 2r]}) = b$). Given a CA F , a configuration $x \in A^{\mathbb{Z}}$ is an *ultimately periodic point* for F if there exists $p \in \mathbb{N}_+, q \in \mathbb{N}$ such that $F^{p+q}(x) = F^q(x)$. If $q = 0$, then x is a *periodic point*, i.e., $F^p(x) = x$. If the set of all periodic points of F is dense in $A^{\mathbb{Z}}$, we say that the CA has *dense periodic orbits (DPO)*. A CA F is *sensitive to initial conditions* (or simply *sensitive*) if there exists $\varepsilon > 0$ such that for any $x \in A^{\mathbb{Z}}$ and any $\delta > 0$ there is an element $y \in A^{\mathbb{Z}}$ such that $d(y, x) < \delta$ and $d(F^n(y), F^n(x)) > \varepsilon$ for some $n \in \mathbb{N}$. A CA F is *positively expansive* if there exists a constant $\varepsilon > 0$ such that for any pair of distinct elements $x, y \in A^{\mathbb{Z}}$ we have $d(F^n(y), F^n(x)) \geq \varepsilon$ for some $n \in \mathbb{N}$. Recall that a CA F is (*topologically*) *transitive* if for any pair of non-empty open sets $U, V \subseteq A^{\mathbb{Z}}$ there exists an integer $n \in \mathbb{N}$ such that $F^n(U) \cap V \neq \emptyset$. A configuration $x \in A^{\mathbb{Z}}$ is an *equicontinuity point* for F if $\forall \varepsilon > 0$ there exists $\delta > 0$ such that for all $y \in A^{\mathbb{Z}}, d(y, x) < \delta$ implies that $\forall n \in \mathbb{N}, d(F^n(y), F^n(x)) < \varepsilon$. A CA is said to be *equicontinuous* if $\forall \varepsilon > 0$ there exists $\delta > 0$ such that for all $x, y \in A^{\mathbb{Z}}, d(x, y) < \delta$ implies that $\forall n \in \mathbb{N}, d(F^n(x), F^n(y)) < \varepsilon$.

3 Asynchronous Cellular Automata

This section introduces *asynchronous cellular automata (ACA)*, surjectivity and injectivity are extended to the new setting.

Let $f : A^{2r+1} \mapsto A$ be a local rule of radius r . We consider the following asynchronous updating for f . At each time t the local rule f is applied on one and only one cell. A sequence $(a_t)_{t>0}$ of integers specifies the index $a_t \in \mathbb{Z}$ of the cell which is updated at the time step $t > 0$.

Definition 1. An ACA is a quadruple (A, f, r, a) where A is a finite alphabet, $f : A^{2r+1} \mapsto A$ is the local rule of radius $r \in \mathbb{N}$ and $a = (a_t)_{t>0}$, with $a_t \in \mathbb{Z}$ is a sequence of cell positions.

Every ACA $\mathcal{A} = (A, f, r, a)$ induces a global behaviour described as follows. For any fixed $k \in \mathbb{Z}$, let $F_k : A^{\mathbb{Z}} \mapsto A^{\mathbb{Z}}$ be the map such that:

$$\forall x \in A^{\mathbb{Z}}, \forall i \in \mathbb{Z}, \quad F_k(x)_i = \begin{cases} f(x_{i-r}, \dots, x_i, \dots, x_{i+r}) & \text{if } i = k \\ x_i & \text{otherwise} \end{cases}$$

For any $t \geq 0$, \mathcal{A} transforms the generic configuration $x \in A^{\mathbb{Z}}$ at the time step t into the configuration $F_{a_t}(x)$ at the time step $t+1$. The dynamics of an ACA is described by the family of functions $\mathcal{T}_{\mathcal{A}} = \{Id, F_{a_1}, F_{a_1} \circ F_{a_2}, \dots, F_{a_1} \circ \dots \circ F_{a_t}, \dots\}$. Remark that all the elements from \mathcal{T} are a continuous maps w.r.t. d . The orbit of a configuration $x \in A^{\mathbb{Z}}$ is the sequence $\gamma_x = (x, F_{a_1}(x), (F_{a_1} \circ F_{a_2})(x), \dots)$ associating with each time step t the configuration $\gamma_x(t) = (F_{a_1} \circ \dots \circ F_{a_t})(x)$ of the ACA at that time.

In many situations we are interested in properties that do not depend on the particular sequence a . In those cases, we will refer to the class $C = (A, f, r)$ of ACA in which the sequence a is not fixed. We will call $C = (A, f, r)$ *uninstantiated ACA* such a class C and we will omit the term *uninstantiated* when it will be clear by the context.

Injectivity and surjectivity are important properties for classical CA. Their adaptation to ACA takes into account the whole family of functions $\{F_k\}_{k \in \mathbb{Z}}$.

Definition 2. An ACA $C = (A, f, r)$, is said to be α -injective if $\forall k \in \mathbb{N}, \forall x, y \in A^{\mathbb{Z}}, x \neq y \Rightarrow F_k(x) \neq F_k(y)$.

In other words, an uninstantiated ACA is injective if every instantiated ACA has all the global functions F_k injective.

Definition 3. An ACA $C = (A, f, r)$, is said to be α -surjective if $\forall k \in \mathbb{N}, \forall x \in A^{\mathbb{Z}}, F_k^{-1}(x) \neq \emptyset$.

In other words, an uninstantiated ACA is surjective if every instantiated ACA has all the global functions F_k surjective.

In classical CA injectivity implies surjectivity [33]. A stronger relation holds between α -injectivity and α -surjectivity:

Proposition 1. Let $C = (A, f, r)$ be an ACA. Then, the following statements are equivalent: i) C is α -injective. ii) C is α -surjective. iii) f is center permutive.

Proof. i) \Leftrightarrow ii). Fix $k \in \mathbb{Z}$. Consider the function $h : A^{2r+1} \mapsto A^{2r+1}$ defined as $h(u) = F_k(x)_{[k-r, k+r]}$ where x is any configuration such that $x_{[k-r, k+r]} = u$.

Then, F_k injective $\Leftrightarrow h$ is injective and F_k surjective $\Leftrightarrow F_k$ is surjective. Since the domain and the codomain of h are equal and of finite cardinality h injective $\Leftrightarrow h$ is surjective.

$i) \Leftrightarrow iii)$. Suppose that C is α -injective and fix $u, v \in A^r$ and $b \in A$. Since h is injective there exists only one $c \in A$ such that $h(ucv) = ubv$ and in particular $f(ucv) = b$. Thus f is center permutive. Vice versa, if f is center permutive, then any block $w = ubv \in A^{2r+1}$ has an unique preimage $h^{-1}(w) = ucv$ for a certain $c \in A$. \square

Remark 1. Unlike classical CA, ACA defined by a rightmost/leftmost permutive local rule are not necessarily α -surjective. As an example consider the local rule $f : \{0, 1\}^3 \mapsto \{0, 1\}$ such that $f(a, b, c) = a$. The rule is leftmost permutive but not center permutive, hence it is not α -surjective.

4 Dynamical Properties of ACA

The adaptation of CA dynamical properties to ACA needs to take into account that there are infinite possible updating sequences. We will distinguish behaviours that can emerge for every sequence from the ones that can only appear for one particular sequence.

A sequence $(s_t)_{t \in \mathbb{N}}$ is ultimately periodic iff there exists a period $n \in \mathbb{N}_+$ and a preperiod $q \in \mathbb{N}$ such that $\forall i \in \mathbb{N} s_{p+q+i} = s_{q+i}$.

The dynamics of an ACA is strictly related to the structure of the updating sequence. In particular the following property holds.

Proposition 2. *Let $C = (A, f, r, a)$ be an ACA. If a is ultimately periodic, then the orbit γ_x of every configuration $x \in A^{\mathbb{Z}}$ is ultimately periodic.*

Proof. If the sequence a is ultimately periodic then the set $K = \{k_1, \dots, k_n\} \subset \mathbb{Z}$ of all the distinct values that appear in a is a finite set. There are at most $|A|^{|K|}$ different configurations in the orbit of a generic $x \in A^{\mathbb{Z}}$. We now associate with every configuration its time step. Since a is eventually periodic with a certain preperiod q and period n , we can consider only $n + q$ different time steps. The possible configurations associated with these time steps form a finite set of cardinality $(n + q)|A|^{|K|}$. So, the dynamics over it has to be ultimately periodic. \square

The classical notion of sensitivity to initial conditions is adapted to both instantiated and uninstanited ACA. Recall that a nonempty family \mathcal{T} of maps is sensitive if there exists $\varepsilon > 0$ such that for any $x \in A^{\mathbb{Z}}$ and any $\delta > 0$, there is an element $y \in A^{\mathbb{Z}}$ such that $d(x, y) < \delta$ and $d(T(x), T(y)) \geq \varepsilon$ for some $T \in \mathcal{T}$.

Definition 4. *An instantiated ACA $C_a = (A, f, r, a)$ is α -sensitive if its family \mathcal{T}_{C_a} is sensitive. An uninstanited ACA $C = (A, f, r)$ is α -sensitive if there exists a sequence $(a_t)_{t > 0}$ such that the instantiated ACA $C_a = (A, f, r, a)$ is sensitive.*

Remark 2. α -sensitivity means that at least one of the instantiated ACA from the class C is α -sensitive to initial conditions. Requiring that all the instantiated ACA are α -sensitive is a meaningless condition. Indeed, choose an integer

$k > 0$ and consider the sequence $a = \{k, k, k, \dots\}$. The orbits of two arbitrary configurations x and y such that $d(x, y) < 2^{-k}$ cannot separate by a distance greater than 2^{-k} .

Recall that a nonempty family \mathcal{T} of maps is said to be expansive if there exists a constant $\varepsilon > 0$ such that for every pair of distinct elements $x, y \in A^{\mathbb{Z}}$, we have $d(T(x), T(y)) \geq \varepsilon$ for some $T \in \mathcal{T}$.

Definition 5. An instantiated ACA $C_a = (A, f, r, a)$ is α -expansive if its family \mathcal{T}_{C_a} is expansive. An uninstantiated ACA $C = (A, f, r)$ is α -expansive if there exists a sequence $a = (a_t)_{t>0}$ such that the instantiated ACA $C_a = (A, f, r, a)$ is α -expansive.

Like classical CA, α -expansivity implies α -sensitivity.

Proposition 3. Let $C = (A, f, r)$ be an ACA with $r > 0$. If f is either leftmost-permutive or rightmost-permutive then C is α -sensitive.

Proof. Suppose that f is rightmost-permutive. Set $\varepsilon = 2^{-r}$ and define the sequence a as $a = (\underbrace{0}, \underbrace{1, 0}, \underbrace{2, 1, 0}, \underbrace{3, 2, 1, 0}, \underbrace{4, 3, 2, 1, 0}, \dots)$. Choose an arbitrary $x \in A^{\mathbb{Z}}$ and $n \in \mathbb{N}$. Let $y \in A^{\mathbb{Z}}$ with $d(x, y) < 2^{-n}$ and $x_{n+1} \neq y_{n+1}$. There exists a first time $t_1 \in \mathbb{N}$ such that $a_{t_1} + r = n + 1$. Since f is rightmost-permutive and $\gamma_x(t_1 - 1)_i = \gamma_y(t_1 - 1)_i$ for $i = \{-n, \dots, n\}$, the smaller cell position in which $\gamma_x(t_1)$ and $\gamma_y(t_1)$ differ is $n - r + 1$. Repeat the previous argument k times with $k = \lfloor \frac{n+1}{r} \rfloor$. In this way, for any $1 \leq j \leq k$ there exists t_j such that $\gamma_x(t_j)$ and $\gamma_y(t_j)$ differ in position $n - jr + 1$ (this is possible since a contains any positive integer infinitely many times). So, at a certain time t_k , the smallest cell position in which $\gamma_{t_k}(x)$ and $\gamma_{t_k}(y)$ differ will be smaller than r . In other words, there exists a time t_k such that $d(\gamma_x(t_k), \gamma_y(t_k)) < 2^{-r} = \varepsilon$.

If f is leftmost permutive the proof is similar by considering the sequence $a' = (0, -1, 0, -2, -1, 0, \dots)$. □

Remark 3. As for classical CA, leftmost/rightmost-permutivity is not a necessary condition for α -sensitivity. For example consider the alphabet $A = \{0, 1, 2\}$, the function $f : A^3 \mapsto A$ defined as $f(x_1, x_2, x_3) = 0$ if $x_3 = 0$, $f(x_1, x_2, x_3) = 1$ otherwise. The ACA $C = (A, f, r, a)$ where $a = (0, 1, 0, 2, 1, 0, \dots)$ is sensitive but f is neither leftmost nor rightmost permutive.

Proposition 4. Let $C = (A, f, r)$ be an ACA with $r > 0$. If f is both leftmost-permutive and rightmost-permutive then C is α -expansive.

Proof. We show that C is α -expansive with expansivity constant $\varepsilon = 2^{-r}$. We exhibit a sequence a'' that interleaves the sequences a and a' as in the proof of Proposition 3. This sequence defines an ACA whose dynamics “pushes” the difference between two arbitrary configurations into the window $[-r, r]$. Such a sequence is $a'' = (0, 0, 1, 0, -1, 0, 2, 1, 0, -2, -1, 0, 3, 2, 1, 0, \dots)$. If the difference between two configurations is at a positive (resp., negative) cell position, then the ACA “pushes” it towards the center by the sub-sequence a (resp., a'). □

Another interesting dynamical property of CA is transitivity. As with sensitivity and expansivity, the notion of transitivity can be adapted to ACA. Recall that a nonempty family \mathcal{T} of maps is said to be transitive if for any pair of non-empty open sets $U, V, T(U) \cap V \neq \emptyset$ for some $T \in \mathcal{T}$.

Definition 6. *An instantiated ACA $C_a = (A, f, r, a)$ is α -transitive if its family \mathcal{T}_{C_a} is transitive. An uninstantiated ACA $C = (A, f, r)$ is α -transitive if there exists a sequence $a = (a_t)_{t>0}$ such that the instantiated ACA $C_a = (A, f, r, a)$ is α -transitive.*

Lemma 1. *Let $C = (A, f, r)$ be an ACA with $r > 0$. Let $B \subseteq \mathbb{Z}$ be a set of positions (called tabu set) and $i \in \mathbb{Z}$. If f is rightmost-permutive (resp., leftmost-permutive) and $\exists j \in \mathbb{Z} \setminus B$ such that $j = i + qr$ (resp., $j = i - qr$) for some $q \in \mathbb{N}_+$ then: $\forall x \in A^{\mathbb{Z}}, \forall b \in A \exists a = (a_1, a_2, \dots)$ and $\exists y \in A^{\mathbb{Z}}$ with $\forall k \in \mathbb{Z} \setminus \{j\} x_k = y_k$ and such that*

$$(F_{a_1} \circ F_{a_2} \circ \dots \circ F_{a_q})(y)_i = b \tag{1}$$

Proof. Suppose that f is rightmost permutive (the leftmost permutive case is similar). Choose $x \in A^{\mathbb{Z}}$ and $b \in A$. We incrementally build the sequence a and the configuration y for a certain $q = n + 1$ using a sequence for $q = n$. If $q = 1$ then let $a = (i, a_2, a_3, \dots)$. Since f is rightmost-permutive for every $b \in A$ there exists $c \in A$ such that $f(x_{i-r}, \dots, x_i, \dots, x_{i+r-1}, c) = b$. Define $y = x_{(-\infty, i+r-1] \setminus C} x_{[i+r+1, +\infty)}$. It is immediate to see that $F_{a_1}(y)_i = b$.

Suppose that we constructed sequences for $q = n$ verifying (1) for position $i' = i + r$. We can also construct sequences for $q = n + 1$. Since have sequences for $q = n$, for all $b' \in A$ we can obtain a sequence $a_{b'} = (a_1, a_2, \dots, a_q, \dots)$ and a configuration $y_{b'}$ such that after n time steps the symbol b' appears in position i' . Since f is rightmost-permutive, for every $b \in A$ there exists a symbol $b' \in A$ such that $f(x_{i-r}, \dots, x_i, \dots, x_{i+r-1}, b') = b$. Then, for $q = n + 1$ and position i , the configurations and the sequence verifying (1) are $y_{b'}$ and $a = (a_1, a_2, \dots, a_q, i, \dots)$, respectively. \square

Proposition 5. *Let $C = (A, f, r)$ be an ACA with $r > 0$. If f is permutive then C is α -transitive.*

Proof. Suppose that f is rightmost permutive (the leftmost permutive case is similar). Choose $x, y \in A^{\mathbb{Z}}$ and $m \in \mathbb{N}$. Let us build a sequence a defining an α -transitive ACA $C_a = (A, f, r, a)$. The sequence a is build incrementally: the first t_0 terms for the case $m = 0$ followed by t_1 terms for $m = 1$ etc. We will construct a configuration x' with $x'_{[-m, m]} = x_{[-m, m]}$ and such that at a certain time $t \gamma_{x'}(t)_{[-m, m]} = y_{[-m, m]}$. For every m we need to set the states of $2m + 1$ cells (from position $-m$ to $+m$) inside $\gamma_{x'}(t)$. Since f is rightmost-permutive we will set the states starting from position $-m$ up to $+m$. Fix $m = n$ and a position $i \in [-m, m]$. Since the sequence is build incrementally, the first h terms of a have already been determined (i.e., the terms for $m - 1, m - 2, \dots, 0$ and for $m = n$ but when the positions inside $[-m, i - 1]$ are considered). Even the configurations x' is build incrementally. A part of x' has already been determined and individuates a corresponding configuration $x'' \in A^{\mathbb{Z}}$. Now set

$B = \{a_k+r \mid k \leq h\} \cup \{-m, \dots, m\}$. Apply Lemma 11 with the tabu set B and the configuration $(F_{a_1} \circ \dots \circ F_{a_t})(x'')$. This gives a sequence $b_1, b_2, \dots, b_{k-1}, i$ such that exists x''' that differs from x'' only in position b_1 and verifying (11). In this way we can fix the term between $h+1$ and $h+k$ of a ($a_{h+1} = b_1, \dots, a_{h+k} = i$) and a sequence x''' such that $d(x, x''') < 2^{-m}$ and $\gamma_{x'''}(t+k)_{[-m, i]} = y_{[-m, i]}$. This gives a sequence a such that for every x and y there exist a configuration x' and a time $t \in \mathbb{N}$ with $d(x, x') < 2^{-m}$ and $d(\gamma_{x'}(t), y) < 2^{-m}$. This means that the ACA $C = (a, f, r)$ is α -transitive. \square

Example 1. As an example consider a rightmost-permutive rule f with $r = 2$. The sequence given by Proposition 5 is:

$$a = \left(\underbrace{0}_{m=0}, \underbrace{1, -1, 2, 0, 3, 1}_{m=1}, \underbrace{4, 2, 0, -2}_{\text{for pos. } -2}, \underbrace{5, 3, 1, -1}_{\text{for pos. } -1}, \underbrace{6, 4, 2, 0}_{\text{for pos. } 0}, \underbrace{7, 5, 3, 1}_{\text{for pos. } 1}, \underbrace{8, 6, 4, 2}_{\text{for pos. } 2}, \dots \right)$$

Remark 4. The terms of the sequence needed to obtain α -sensitivity and α -transitivity have to be unbounded. Indeed for α -sensitivity, we need to consider differences that can be arbitrarily far from the center. For α -transitivity, the reason is similar: to generate a central block of an arbitrary size it is necessary to have a sequence containing arbitrarily far positions.

Another important notion regarding the dynamics of a CA is DPO. To define α -DPO we need the notion of periodic point for an ACA.

Definition 7. Let $C_a = (A, f, r, a)$ be an instantiated ACA. A point $x \in A^{\mathbb{Z}}$ is called periodic if there exists $p \in \mathbb{N}_+$ such that for all $n \in \mathbb{N}$, $\gamma_x(n) = \gamma_x(n+p)$.

Definition 8. Let $C_a = (A, f, r, a)$ be an instantiated ACA. C_a has α -DPO if its set of periodic points is dense in $A^{\mathbb{Z}}$. An uninstantiated ACA $C = (A, f, r)$ has α -DPO if there exists a sequence $a = (a_t)_{t>0}$ such that the instantiated ACA $C_a = (A, f, r, a)$ has α -DPO.

Proposition 6. Let $C = (A, f, r)$ be an ACA. If C is α -surjective then it has α -DPO.

Proof. Consider the sequence $a = (k, k, k, \dots)$ for a generic $k \in \mathbb{N}$. Fix $x_{(-\infty, k-1]}$ and $x_{[k+1, +\infty)}$ and consider the possible values that $\gamma_x(t)_k$ can assume. For every $b = F_k(x)_{[k+1, +\infty)}$ there exists exactly one $a \in A$ such that $F_k(x_{(-\infty, k-1]} a x_{[k+1, +\infty)}) = b$. Because only the cell in position k changes, we cannot have neither an aperiodic orbit (indeed the number of states in the orbit of x is finite) nor an ultimately periodic orbit (indeed every state has exactly one preimage). This means that every orbit is periodic, hence the ACA has α -DPO. \square

Remark 5. There exists an α -injective uninstantiated ACA that is α -surjective, in other words, an ACA that is α -sensitive, α -transitive and α -DPO. As an example consider the set $A = \{0, 1\}$ and the rule $f : \{0, 1\}^3 \mapsto \{0, 1\}$ defined as $f(x_{i-1}, x_i, x_{i+1}) = x_{i-1} \text{ xor } x_i$. The function is both leftmost-permutive and center-permutive. Then, by Proposition 13.5.6 the ACA $(\{0, 1\}, f, 1)$ is α -surjective, α -sensitive, α -transitive and α -DPO.

Proposition 7. *Let $C = (A, f, r, a)$ be an ACA which has α -DPO. If all the F_{a_t} are not the identity, then a is a bounded sequence.*

Proof. For the sake of argument, suppose that an unbounded sequence $a = (a_1, a_2, \dots)$ can be a sequence of a regular CA $C = (A, f, r, a)$. Since a is unbounded there is an infinite set B of positions inside a appear either in a finite number of time or an infinite number of time but in an aperiodic way inside the sequence. Since C has α -DPO, for every $i \in B$ $F_i(x) = x$ for all $x \in A^{\mathbb{Z}}$ (otherwise we would have a state in position i that would not reappear in a periodic manner). In fact, consider a generic $i \in B$. Because F_i is not the identity there exists $x \in A^{\mathbb{Z}}$ such that $F_i(x) \neq x$. Now pick $\delta = 2^{-\max\{|i+r|, |i-r|\}}$. Every $y \in A^{\mathbb{Z}}$ such that $d(x, y) < \delta$ has the property that $F_i(y) \neq y$. Since the position i is not changed in a periodic manner, all the y are not periodic. This means that the sequence a cannot be the sequence of the given ACA. \square

Corollary 1. *There are no instantiated ACA $C = (A, f, r, a)$ which are both α -sensitive and α -transitive and have α -DPO.*

Proof. The sequence a need to be unbounded to have sensitivity and transitivity but bounded to give α -DPO. \square

The previous Corollary states that it is impossible to have an instantiated ACA $C = (A, f, r, a)$ that respects the traditional definition of Devaney chaos [18].

An other important property of classical CA is equicontinuity. Recall that a family of functions \mathcal{T} is said to be equicontinuous at a point $x \in A^{\mathbb{Z}}$ if for all $\varepsilon > 0$ there exists $\delta > 0$ such that for all $y \in A^{\mathbb{Z}}$, $d(x, y) < \delta$ implies that $d(T(x), T(y)) < \varepsilon$ for all $T \in \mathcal{T}$. The family is equicontinuous if it is equicontinuous at every point.

Definition 9. *An instantiated ACA $C_a = (A, f, r, a)$ is α -equicontinuous if its family \mathcal{T}_{C_a} is equicontinuous. An uninstantiated ACA $C = (A, f, r)$ is α -equicontinuous if for every sequence $a = (a_t)_{t>0}$ the instantiated ACA $C_a = (A, f, r, a)$ is α -equicontinuous.*

Our extension of this property to uninstantiated ACA use a different technique than the one used for α -sensitivity, α -DPO and α -transitivity. In fact, the existence of a sequence $a = (a_1, a_2, \dots)$ such that the resulting family \mathcal{T}_{C_a} is equicontinuous is a condition that is always true. Consider the sequence $a = (0, 0, \dots)$ and $\delta = 2^{-r}$. It is immediate that the resulting family is equicontinuous. Therefore we need to use a different definition for α -equicontinuity.

From the previous definition follow immediately that every α -equicontinuous ACA cannot also be α -sensitive. Some examples of α -equicontinuous ACA are the one defined by the identity function ($F_k = Id$ for all k) and the ones defined by a local function in the form $f(u) = b$ for every $u \in A^{2r+1}$.

In classical CA equicontinuity is equivalent to ultimate periodicity [30], but in ACA this is not true. As an examples consider $A = \{0, 1\}$ and $f : A \rightarrow A$ defined as $f(x) = 0$ for all x . The resulting CA is α -equicontinuous, but the dynamics of at least one point with at least one sequence is not ultimately periodic. As an example consider $x = 1^\infty$ and $a = (0, 1, 2, 3, 4, \dots)$. The dynamics of x with the sequence a is obviously aperiodic.

5 Conclusions

In this work a new kind of asynchronous cellular automata has been introduced. The classical notions of surjectivity, injectivity, sensitivity, transitivity, DPO and equicontinuity has been defined for ACA. The notion of permutivity has a central role since it is strictly linked to these properties. Further investigations involve the study of other relationships between the properties of ACA and an adaptation of other notions of classical CA to the asynchronous case.

References

1. Acerbi, L., Dennunzio, A., Formenti, E.: Shifting and lifting of cellular automata. In: Cooper, S.B., Löwe, B., Sorbi, A. (eds.) CiE 2007. LNCS, vol. 4497, pp. 1–10. Springer, Heidelberg (2007)
2. Acerbi, L., Dennunzio, A., Formenti, E.: Conservation of some dynamical properties for operations on cellular automata. *Theoretical Computer Science* 410, 3685–3693 (2009)
3. Alarcon, T., Byrne, H., Maini, P.: A cellular automaton model for tumour growth in inhomogeneous environment. *Journal of Theoretical Biology* 225(2), 257–274 (2003)
4. Bersini, H., Detours, V.: Asynchrony induces stability in cellular automata based models. In: Proc. of Artificial Life IV, pp. 382–387. MIT Press, Cambridge (1994)
5. Buvel, R., Ingerson, T.: Structure in asynchronous cellular automata. *Physica, D* 1, 59–68 (1984)
6. Cattaneo, G., Dennunzio, A., Formenti, E., Provillard, J.: Non-uniform cellular automata. In: Dediu, A.H., Ionescu, A.-M., Martín-Vide, C. (eds.) LATA 2009. LNCS, vol. 5457, pp. 302–313. Springer, Heidelberg (2009)
7. Cattaneo, G., Dennunzio, A., Margara, L.: Solution of some conjectures about topological properties of linear cellular automata. *Theor. Comput. Sci.* 325(2), 249–271 (2004)
8. Cervelle, J., Dennunzio, A., Formenti, E.: Chaotic behavior of cellular automata. In: Meyers, B. (ed.) *Mathematical basis of cellular automata. Encyclopedia of Complexity and System Science*, vol. 1, pp. 978–989. Springer, Heidelberg (2009)
9. Chaudhuri, P., Chowdhury, D., Nandi, S., Chattopadhyay, S.: *Additive Cellular Automata Theory and Applications*, vol. 1. IEEE Press, New York (1997)
10. Chopard, B.: Modelling physical systems by cellular automata. In: Rozenberg, G., et al. (eds.) *Handbook of Natural Computing: Theory, Experiments, and Applications*. Springer, Heidelberg (to appear, 2010)
11. Dennunzio, A., Di Lena, P., Formenti, E., Margara, L.: On the directional dynamics of additive cellular automata. *Theoretical Computer Science* 410, 4823–4833 (2009)
12. Dennunzio, A., Formenti, E.: Decidable properties of 2d cellular automata. In: Ito, M., Toyama, M. (eds.) DLT 2008. LNCS, vol. 5257, pp. 264–275. Springer, Heidelberg (2008)
13. Dennunzio, A., Formenti, E.: 2d cellular automata: new constructions and undecidability (2010) (preprint)
14. Dennunzio, A., Formenti, E., Kůrka, P.: Cellular automata dynamical systems. In: Rozenberg, G., et al. (eds.) *Handbook of Natural Computing: Theory, Experiments, and Applications*. Springer, Heidelberg (to appear, 2010)

15. Dennunzio, A., Guillon, P., Masson, B.: Stable dynamics of sand automata. In: Ausiello, G., Karhumäki, J., Mauri, G., Ong, C.-H.L. (eds.) IFIP TCS. IFIP, vol. 273, pp. 157–169. Springer, Heidelberg (2008)
16. Dennunzio, A., Guillon, P., Masson, B.: Sand automata as cellular automata. *Theor. Comput. Sci.* 410(38-40), 3962–3974 (2009)
17. Dennunzio, A., Masson, B., Guillon, P.: Sand automata as cellular automata. *Theoretical Computer Science* 410, 3962–3974 (2009)
18. Devaney, R.L.: *An Introduction to chaotic dynamical systems*, 2nd edn. Addison-Wesley, Reading (1989)
19. Di Lena, P., Margara, L.: Computational complexity of dynamical systems: the case of cellular automata. *Information and Computation* 206, 1104–1116 (2008)
20. Di Lena, P., Margara, L.: On the undecidability of the limit behavior of cellular automata. *Theoretical Computer Science* 411, 1075–1084 (2010)
21. Farina, F., Dennunzio, A.: A predator-prey cellular automaton with parasitic interactions and environmental effects. *Fundamenta Informaticae* 83, 337–353 (2008)
22. Fatès, N., Morvan, M.: An experimental study of robustness to asynchronism for elementary cellular automata. *Complex Systems* 16(1), 1–27 (2005)
23. Fatès, N., Morvan, M., Schabanel, N., Thierry, E.: Fully asynchronous behaviour of double-quiescent elementary cellular automata. *Theoretical Computer Science* 362, 1–16 (2006)
24. Fatès, N., Regnault, D., Schabanel, N., Thierry, E.: Asynchronous behaviour of double-quiescent elementary cellular automata. In: Correa, J.R., Hevia, A., Kiwi, M. (eds.) LATIN 2006. LNCS, vol. 3887, pp. 455–466. Springer, Heidelberg (2006)
25. Frisch, U., Hasslacher, B., Pomeau, Y.: Lattice-gas automata for the navier–stokes equation. *Physical Review Letters* 56, 1505–1508 (1986)
26. Fukš, H.: Non-deterministic density classification with diffusive probabilistic cellular automata. *Physical Review E* 66(2) (2002)
27. Fukš, H.: Probabilistic cellular automata with conserved quantities. *Nonlinearity* 17(1), 159–173 (2004)
28. Kari, J.: Tiling problem and undecidability in cellular automata. In: Meyers, B. (ed.) *Mathematical basis of cellular automata*. Encyclopedia of Complexity and System Science, pp. 9158–9172. Springer, Heidelberg (2009)
29. Kier, L., Seybold, P., Cheng, C.-K.: *Modeling Chemical Systems using Cellular Automata*. Springer, Heidelberg (2005)
30. Kůrka, P.: Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory & Dynamical Systems* 17, 417–433 (1997)
31. Kůrka, P.: Topological dynamics of one-dimensional cellular automata. In: Meyers, B. (ed.) *Mathematical basis of cellular automata*. Encyclopedia of Complexity and System Science, pp. 2232–2242. Springer, Heidelberg (2009)
32. Lumer, E., Nicolis, G.: Synchronous versus asynchronous dynamics in spatially distributed systems. *Physica, D* 71, 440–452 (1994)
33. Maruoka, A., Kimura, M.: Injectivity and surjectivity for parallel maps for CA. *J. Comp. and Sys. Sci* 18, 47–64 (1979)
34. Regnault, D.: Abrupt behaviour changes in cellular automata under asynchronous dynamics. In: *Electronic Proc. of 2nd European Conference on Complex Systems, ECCS*, Oxford, UK (2006)
35. Regnault, D., Schabanel, N., Thierry, E.: Progresses in the analysis of stochastic 2d cellular automata: A study of asynchronous 2d minority. *Theoretical Computer Science* 410, 4844–4855 (2009)
36. Schönfisch, B., de Roos, A.: Synchronous and asynchronous updating in cellular automata. *BioSystems* 51, 123–143 (1999)

A Study on the Automatic Generation of Asynchronous Cellular Automata Rules by Means of Genetic Algorithms

Andrea Valsecchi, Leonardo Vanneschi, and Giancarlo Mauri

Complex Systems and Artificial Intelligence (C.S.A.I.) Research Center
Department of Informatics, Systems and Communication (D.I.S.Co.)
University of Milano-Bicocca, Milan, Italy
{valsecchi,vanneschi,mauri}@disco.unimib.it

Abstract. We present a framework based on genetic algorithms to automatically generate cellular automata rules under four different asynchronous update models (fixed random sweep, random new sweep, clock and independent random ordering). We consider four different rules (18, 56, 110 and 180) with well known dynamics under synchronous update scheme. We try to reconstruct the same dynamics by means of a genetic algorithm using asynchronous update schemes. We show that in many cases it is impossible, by means of an asynchronous update scheme, to perfectly reconstruct these dynamics. Nevertheless, we show that the genetic algorithm finds the rules that more closely approximate the target behavior and the dynamics of the rules found by the genetic algorithm are rather similar to the target ones. In particular, we can always recognize a similar pattern and we can also identify some differences in small details, which can be minimal (as for rule 18) or rather visible (as for rule 110). This paves the way to a deeper investigation on this track: does using asynchronous updates allow us to find more stable rules, i.e. rules that are less affected by noise, and thus do not overfit training data? This question remains open and answering it is one of the main goals of our current research.

1 Introduction

Cellular automata (CAs) are discrete dynamical systems where several cells, characterized by a state, evolve according to the states of their neighboring cells. They have been studied theoretically for years due to their architectural simplicity and the wide spectrum of behaviors they are capable of [6,32]. CAs can perform universal computation and, given that their time evolution can be complex, they are often used to model complex phenomena. Nevertheless, many CAs show interesting dynamical properties such as fixed points and cyclic attractors, which make them an attracting field for theoretical studies.

Historically, CAs have been studied under synchronous dynamics where all the cells update at the same time, given that this makes them easier to formalize and their properties easier to study. Nevertheless, many real-life phenomena

clearly have an asynchronous nature; it is the case for instance of biological systems, systems of chemical reactions, crowds and many other complex systems. Given the difficulty of the formalization of such a variant of CAs, relatively few studies of asynchronous CAs have appeared to date. For instance, some studies focus on a CA model evolving synchronously but where some random errors can occur, like in [31], where Toom defines a 2D CA capable of remembering one bit of information in presence of random error. This result is used in [12] to develop a 3D CA capable of reliable computation. Later on the existence of a 1D CA exhibiting the same reliability was proven in [11]. In [1] the authors try to apply the mean field approach on a probabilistic model of CAs and show that complex behaviors cannot be explained by this method. Several empirical studies have shown that the behavior of CAs changes drastically under asynchronous dynamics [3,5,8,18,24]. Only few theoretical results are known. Mainly, either they concern specific CAs or show that it is difficult to describe the global behavior of CAs under asynchronous updates [10,11,10,22,9,23].

In the absence of solid theoretical basis, machine learning [21] is a possibility for trying to find the rule of an asynchronous CA that exhibits given dynamics. The idea of automatically generating a rule that performs a given computational task has been widely explored in literature (see for instance [29,16,13]). However, these and many other studies generally deal with synchronous CAs, and even for them the task of automatically determining rules is generally considered a very hard one. In addition to the often huge size of the search space (i.e. the set of all the possible rules), another factor that contributes to make this problem hard is the fact that the behavior of a CA rule is not easy to predict just by looking at the syntactical representation of the rule itself, and two rules with extremely similar representations can result both in almost identical or largely different global dynamics [33]. These factors, among the others, have favored the development of the use of Genetic Algorithms (GAs) [15,14] to explore the rule space, given their implicit parallelism and their ability to search difficult and complex spaces. Potential solutions (or individuals) evolved by the GAs are CA transition rules represented as strings of characters as in [33] (this representation is also used here and presented in Section 3).

A wide amount of literature exists about the use of GAs for evolving synchronous CA rules. A review can be found in [20]. The work of Sipper and colleagues represents a noteworthy contribution to the field (see for instance [25,26,27]). Furthermore, GAs have also been used for pattern recognition in CA and complex systems [2]. In [13,19] the use of GAs was proposed to design Multiple Attractor CA to perform pattern classification.

However, very few studies have appeared to date dealing with the problem of evolving rules for asynchronous CAs, noteworthy exceptions being represented by [28,30,17] where asynchronous rules are evolved for the well known density task. In this paper we approach this problem, presenting a GA framework to evolve asynchronous CA rules, using different models of asynchronous update strategy and trying to evolve rules for different dynamics than the ones of the density task.

Even though we know that many readers may not be expert in the field of GAs, the space limitation prevents us from including in this paper an introduction to GAs. The interested reader is referred to [15][14]. The rest of the paper is organized as follows: Section 2 presents the studied models of asynchronous update; in Section 3 we show how our experimental study is organized and we discuss the experimental setting; in Section 4 we discuss the obtained results; finally Section 5 concludes the paper.

2 Asynchronous CA Models and Case Studies

In [28] and in [30] the authors evolve rules for three different models of asynchronous CAs: *independent random ordering*, *fixed random sweep* and *random new sweep* and they concentrate on the well known and widely studied density task, where the configuration of the CA must relax to a fixed-point pattern of all 1s if the initial configuration of states contains more 1s than 0s and to a fixed-point pattern of all 0s otherwise.

In this paper we also evolve CA rules for the three previously considered asynchronous models (independent random ordering, fixed random sweep and random new sweep), but we also consider another schema of asynchronous update that has been called *clock* in [7], and that will be defined shortly in this paper. To the best of our knowledge this paper represents the first attempt of evolving CA rules using this asynchronous model. Furthermore, instead of concentrating on the density task, we consider a set of different CA dynamics, typically induced by well-known rules under synchronous update scheme, and we try to evolve the rules for asynchronous CAs that more closely approximate those dynamics. This extends the previous studies concerning the evolution of rules for asynchronous CAs to a set of tasks that had never been considered before.

In this section, we first describe the four considered models of asynchronous CAs and successively we present the dynamics that we want to approximate with the rules found with our GA framework.

The models of asynchronous CAs that we consider are (see [7] for a more detailed and complete introduction to these models): (1) *Fixed random sweep*: at each iteration all CA cells are updated in a pre-defined random order that stays the same at each subsequent iteration. (2) *Random new sweep*: at each iteration all CA cells are updated in a random order and this order changes at each iteration. (3) *Clock*: A frequency f_i is associated to each cell c_i of the CA. For each $i = 1, \dots, n$ (with n equal to the number of cells in the CA), c_i is updated at every f_i iterations. (4) *Independent random ordering*: at each iteration, n updates are done (where n is the number of cells) and the updated cell is chosen randomly with uniform probability at each step. This choice is done with replacement, in the sense that once a cell has been chosen, at the subsequent step it can be chosen again.

The dynamics that we want to reconstruct are the ones induced by the following well-known elementary rules under synchronous update scheme: 18, 56, 110, 180. These rules and their dynamics under synchronous update scheme are documented for instance in [6][32] (and a graphical representation of their dynamics

under synchronous update scheme are presented in the upper row of Figure 4). Here we want to find the rules that better approximate these dynamics under the asynchronous update schemes presented above.

3 Experimental Settings

In the GA, a rule e is represented as a binary string of length 2^{2r+1} where r is the radius of the neighborhood of the cells considered for applying the update rule. If j is the number obtained interpreting the neighborhood of a cell c as a binary number, the new state of c according to e is the j^{th} character of this string (this is the same method for representing CA rules as GA potential solutions as in [33]). The length of a configuration of the CA was set to 101 bits. To rate the performance of a rule e in mimicing the dynamics of a target rule t under an asynchronous update schema s , we consider the list of 50 consecutive configurations obtained using the rule e when the starting configuration i consist of a 1 in the center surrounded by 0s. For each configuration in the list, we compute the Hamming distance from the corresponding configuration in the list of the target rule, then the values are summed up and scaled to fit in the interval $[0, 1]$. We used the following formula:

$$f(e, s, t) = \frac{\sum_{c=1}^{50} H(e_s^c(i), t^c(i))}{50 \cdot 101}$$

When a non deterministic update schema is used (i.e. random new sweep or independent random ordering), the error at each step is averaged over 30 repetitions:

$$f'(e, s, t) = \frac{\sum_{c=1}^{50} \sum_1^{30} H(e_s^c(i), t^c(i))}{50 \cdot 101 \cdot 30}$$

The tests were divided into two series according to the values of the radius of the neighborhood of candidate solutions. In the first series we evolve elementary rules (radius 1), therefore individuals have length 8 bits and the search space is composed by 256 elements; the GA uses standard mutation and crossover, fitness-proportionate selection, a small population and a low number of generations. In the second series, even though the target dynamics remain the same (and they can be obtained by synchronous updates using radius 1), we have tried to approximate them using rules of radius 2 and then there are $2^{32} \approx 4 \cdot 10^9$ candidate solutions; in the GA the size of the population, mutation rate and the maximum number of generations are increased accordingly. Furthermore, in all our experiments we have used elitism, i.e. we copy the best individual into the next population at each generation. The values of the parameters used by the GA are shown in Table 1.

To measure the performance of the GA, we executed 100 independent runs for each target rule and update schema. For comparison, also the results obtained using the synchronous schema are included.

Table 1. Values of parameters used by the Genetic Algorithm in the two series of tests

	Series I	Series II
Individual length	8	32
Population size	10	100
Generations	15	50
Selection	proportionate	tournament, size 5
Crossover rate	0.9	0.9
Mutation rate	0.125	0.03125
Elitism	Yes	Yes

4 Experimental Results

Before presenting the results that show how we have been able to reconstruct the target dynamics, it is natural to answer a question: is it possible to reconstruct these dynamics in a perfect way using an asynchronous update scheme? In order to give a hint on how to answer this question, Figure 1 reports the fitness of all the potential solutions (CA rules) in the search space in case of radius 1 (thus we have 256 possible solutions), when independent random ordering has been used as the model of asynchronous updates. Figure 1(a) (respectively 1(b), 1(c) and 1(d)) reports the results for update rule 18 (respectively 56, 110 and 180).

These figures should give the reader an idea of the shape of the fitness landscape of these problems. It must be clear that this is just a rough approximation, given that solutions are sorted on the horizontal axis according to the number their binary code represents, and this has in principle no relationship with the neighborhoods induced by the genetic operators employed by the GA (we could informally say that the landscapes represented in Figure 1 are not the landscapes “seen” by the GA). Nevertheless, we believe that these histograms can give some useful insight on the search spaces we are studying. For instance, it is possible to remark that no solution in the search space has a fitness value exactly equal to 1. This means that it is *impossible* (at least for the cases we are showing, i.e. radius 1, independent random ordering update and rules 18, 56, 110 and 180 as target) to obtain, by means of an asynchronous update, exactly the same behavior that is obtained using the traditional synchronous update. In particular, we can see that for rules 18 and 110 it is slightly larger than 0.85, while for rules 56 and 180 it is approximately equal to 0.95. Even though the histograms are not reported here for lack of space, we can confirm that, in case of radius 1, the same trend is visible also for all the other studied models of asynchronous update.

Furthermore, it is interesting to point out that the target rules themselves (i.e. the ones that would have a perfect fitness if the synchronous update was used) have a fitness value smaller than 1 when the asynchronous update models are used. In particular, in case of independent random ordering, rule 18 has a fitness value equal to 0.457 (which is remarkably worse than the best possible fitness, which is around 0.85), rule 56 has a fitness value equal to 0.987, rule 110 has a fitness value equal to 0.863 and rule 180 has a fitness value equal to 0.968.

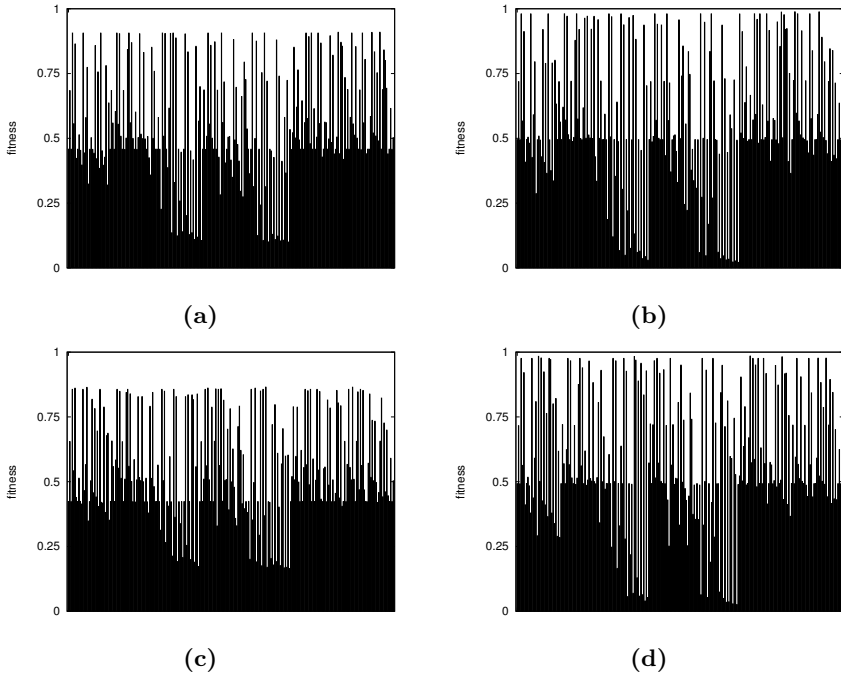


Fig. 1. The fitness of all the candidate rules for the *independent random ordering* asynchronous model and radius 1. Histogram (a) (respectively (b), (c) and (d)) reports the results for update rule 18 (respectively 56, 110 and 180)

As a consequence, it is only normal that our GA system returns rules that are different from the target synchronous one.

The experimental results of the GA system are reported in Figures 2 and 3.

Figure 2 refers to the first series of experiments, while Figure 3 refers to Series II. Each figure shows the mean best fitness and the average number of generations needed by the GA to terminate (either by finding an optimal solution or reaching the maximum number of generations). A series of bars refers to the tests performed using the corresponding target rule and each bar is relative to a different update model. We can see that in all cases all update models obtain approximately the same performance (the differences between the mean best fitness values obtained using the different update rules, in fact, are not statistically significant). Furthermore, if we compare the results in Figure 2 with the histograms in Figure 1, we can observe that, at least for radius 1, our GA system has allowed us to find the solutions with the best possible fitness value in the whole search space. Finally, we point out that if we compare the results in Figure 2 with the ones in Figure 3, we can see that in the case of radius 2 the results obtained using the different update models are very similar to the ones obtained in the case of radius 1. This hints that in the case of radius 2 the search space might have a similar shape to the one of radius 1 reported in Figure 1. In

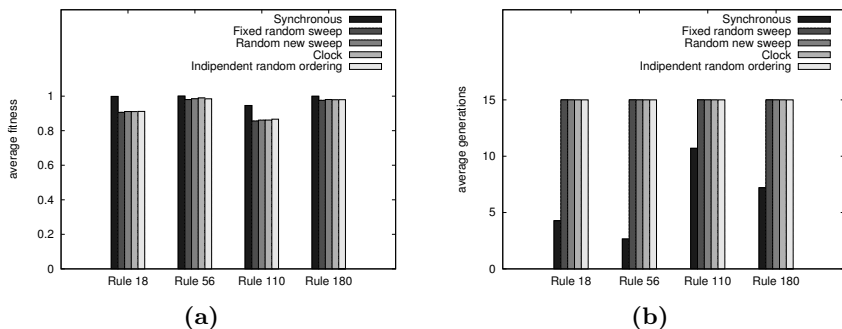


Fig. 2. Experimental results obtained evolving rules having *radius one* using different update schemes. Each series of bars correspond to a different target rule. (a): mean best fitness at termination; (b): average number of generations needed to terminate (either because an optimal solution has been found or because the maximum number of generations has been reached).

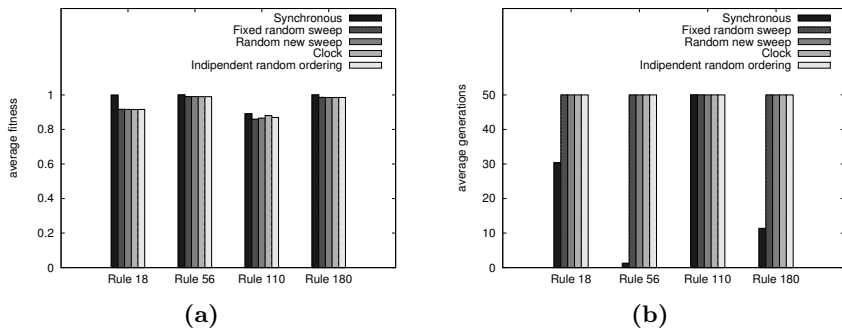


Fig. 3. Experimental results obtained evolving rules having *radius two* using different update schema's. Each series of bars correspond to a different target rule.(a): mean best fitness at termination; (b): average number of generations needed to terminate (either because an optimal solution has been found or because the maximum number of generations has been reached).

particular, we believe that also for radius 2 no individual in the search space has a fitness value exactly equal to 1.

The fact that no solution in the search space has a fitness value equal to 1 implies that no rule using an asynchronous update model can return exactly the same temporal behavior (dynamics) as the target rule executed with the traditional synchronous update. Thus a question arises naturally: what dynamics the rules found by our GA system have, when executed with the considered update models? And what is the relationship between this temporal behavior and the well known behavior shown by the target rule under the traditional synchronous update? Figure 4 answers this question for independent random ordering (but the other considered models, although not reported here for lack

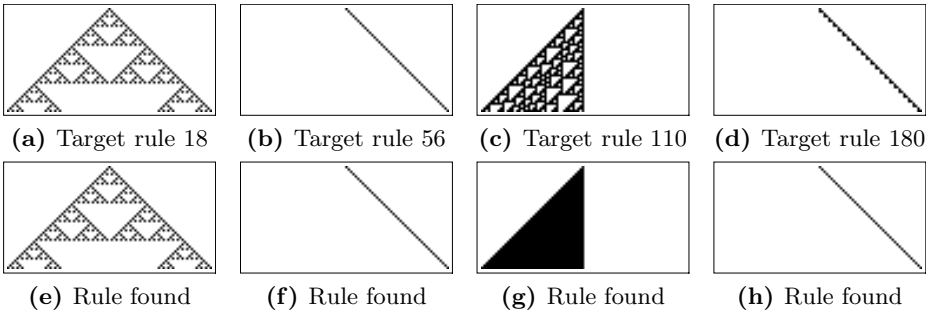


Fig. 4. The upper row reports the target dynamics for rules 18, 56, 110 and 180 under synchronous update. The lower row reports the dynamics produced by the best rule found by our GA framework under independent random ordering update.

of space, allow us to reconstruct very similar dynamics). The upper row reports the dynamics of the target rule under the traditional synchronous update. The lower row reports the dynamics of the best solution found by our GA system over all the runs we have executed using independent random ordering update. As we can see, the dynamics of rule 18, 56 and 180 are reconstructed rather faithfully (although with some small differences compared to the traditional schema) by the rule found by the GA system. For rule 110, the general pattern has been correctly reconstructed, but many details of the target rule are not replicated when using independent random ordering (in particular the white spaces inside the black triangle have not been reconstructed).

5 Conclusions

A framework based on Genetic Algorithms (GAs) to automatically generate Cellular Automata (CA) rules under four different asynchronous update models (fixed random sweep, random new sweep, clock and independent random ordering) is presented in this paper. The process we have followed consisted in considering four CA rules (rules 18, 56, 110 and 180) whose dynamics under synchronous update are well known, and looking for the rule that more closely simulates those dynamics using the different studied asynchronous models by means of our GA framework.

First of all, we have shown that perfectly reconstructing the target behavior by means of an asynchronous update model is often impossible. In fact, at least for the restricted case of radius 1, no candidate rule scores a perfect fitness. Furthermore, our GA system is able to find the rules that have the best fitness values among all the candidate ones, and no significant difference is observed among the different asynchronous models. Analyzing the dynamics of those rules, we have remarked interesting similarities with the target, in the sense that the general spatial pattern is reconstructed, but some details are not. We deduce that asynchronous update models can roughly approximate the dynamics of synchronous

ones. Our current research activity consists in the analysis of the generalization ability of our GA framework: does the fact that we roughly approximate the target behavior, often abstracting from some details, allows us to obtain better generalization ability? Can the amount of information that is not reconstructed by our framework be considered as “noise”, or as information that causes the overfitting of training data? Can we say that our GA framework is able to recognize dynamic patterns? All these questions are important and open the door to a new and stimulating research activity. Furthermore, we also plan to investigate one more asynchronous update scheme introduced in [4]: a stochastic scheme in which each cell has a certain probability to be updated at every time step. In other words, at every time step a random collection of cells is updated in this scheme. This scheme is a generalization of the synchronous updating scheme, which corresponds to the update probability 1. This scheme offers the possibility to study updating schemes corresponding to a continuum of probabilities ranging from just above 0 (very asynchronous) to 1 (completely synchronous).

References

1. Balister, P., Bollobás, B., Kozma, R.: Large deviations for mean fields models of probabilistic cellular automata. *Random Structures and Algorithms* 29(1), 339–415 (2006)
2. Bandini, S., Vanneschi, L., Wuensche, A., Bahgat Shehata, A.: A neuro-genetic framework for pattern recognition in complex systems. *Fundamenta Informaticae* 87(2) (2008)
3. Bersini, H., Detours, V.: Asynchrony induces stability in cellular automata based models. In: *Proceedings of Artificial Life IV*, pp. 382–387. MIT Press, Cambridge (1994)
4. Blok, H.J., Bergersen, B.: Synchronous versus asynchronous updating in the game of life. *Phys. Rev. E* 59, 3876–3879 (1999)
5. Buvel, R.L., Ingerson, T.E.: Structure in asynchronous cellular automata. *Physica D* 1(1), 59–68 (1984)
6. Chopard, B., Droz, M.: *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, Cambridge (1998)
7. Cornforth, D., Green, D.G., Newth, D.: Ordered asynchronous processes in multi-agent systems. *Physica D: Nonlinear Phenomena* 204(1-2), 70–82 (2005)
8. Fatés, N., Morvan, M.: An experimental study of robustness to asynchronism for elementary cellular automata. *Complex Systems* 16(1), 1–27 (2005)
9. Fatés, N., Regnault, D., Schabanel, N., Thierry, E.: Asynchronous behaviour of double-quiescent elementary cellular automata. In: Correa, J.R., Hevia, A., Kiwi, M. (eds.) *LATIN 2006*. LNCS, vol. 3887, pp. 455–466. Springer, Heidelberg (2006)
10. Fukás, H.: Probabilistic cellular automata with conserved quantities. *Nonlinearity* 17(1), 159–173 (2004)
11. Gács, P.: Reliable cellular automata with self-organization. *Journal of Statistical Physics* 103(1/2), 45–267 (2001)
12. Gács, P., Reif, J.: A simple three-dimensional real-time reliable cellular array. *Journal of Computer and System Sciences* 36(2), 125–149 (1988)
13. Ganguly, N., Maji, P., Dhar, S., Sikdar, B.K., Chaudhuri, P.P.: Evolving Cellular Automata as Pattern Classifier. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) *ACRI 2002*. LNCS, vol. 2493, pp. 56–68. Springer, Heidelberg (2002)

14. Goldberg, D.E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading (1989)
15. Holland, J.H.: *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor (1975)
16. Ibarra, O.H., Palis, M.A., Kim, S.M.: Fast parallel language recognition by cellular automata. *Theoretical Computer Science* 41(2-3), 231–246 (1985)
17. Jeanson, F.: Evolving asynchronous cellular automata for density classification. In: Bullock, S., Noble, J., Watson, R., Bedau, M.A. (eds.) *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, pp. 282–288. MIT Press, Cambridge (2008)
18. Lumer, E.D., Nicolis, G.: Synchronous versus asynchronous dynamics in spatially distributed systems. *Physica D* 71(1), 440–452 (1994)
19. Maji, P., Shaw, C., Ganguly, N., Sikdar, B.K., Pal Chaudhuri, P.: Theory and application of cellular automata for pattern classification. *Fundam. Inf.* 58(3-4), 321–354 (2003)
20. Mitchell, M., Crutchfield, J.P., Das, R.: Evolving cellular automata with genetic algorithms: A review of recent work. In: *Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA 1996)* (1996)
21. Mitchell, T.: *Machine Learning*. McGraw Hill, New York (1996)
22. Regnault, D.: Abrupt behavior changes in cellular automata under asynchronous dynamics. In: *Proceedings of 2nd European Conference on Complex Systems (ECCS)*, Oxford, UK (2006)
23. Regnault, D., Schabanel, N., Thierry, É.: Progresses in the analysis of stochastic 2d cellular automata: a study of asynchronous 2d minority. In: Kučera, L., Kučera, A. (eds.) *MFCS 2007. LNCS*, vol. 4708, pp. 320–332. Springer, Heidelberg (2007)
24. Schönfish, B., de Ross, A.: Synchronous and asynchronous updating in cellular automata. *BioSystems* 51(1), 123–143 (1999)
25. Sipper, M.: Non-Uniform Cellular Automata: Evolution in Rule Space and Formation of Complex Structures. In: Brooks, R.A., Maes, P. (eds.) *Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems (Artificial Life IV)*, pp. 394–399. MIT Press, Cambridge (1994)
26. Sipper, M.: Co-evolving non-uniform cellular automata to perform computations. *Physica D* 92(3-4), 193–208 (1996)
27. Sipper, M.: *Evolution of parallel cellular machines: The Cellular Programming Approach*. Springer, New York (1997)
28. Sipper, M., Tomassini, M., Capcarrere, M.S.: Evolving asynchronous and scalable non-uniform cellular automata. In: *Proceedings of International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA 1997)*, pp. 67–71. Springer, Heidelberg (1997)
29. Smith III, A.R.: Real-time language recognition by one-dimensional cellular automata. *Journal of Computer and System Sciences* 6(3), 233–253 (1972)
30. Tomassini, M., Venzi, M.: Artificially evolved asynchronous cellular automata for the density task. In: *ACRI 2001: Proceedings of the 5th International Conference on Cellular Automata for Research and Industry*, London, UK, pp. 44–55. Springer, Heidelberg (2002)
31. Toom, A.: Stable and attractive trajectories in multicomponent systems. *Advances in Probability* 6(1), 549–575 (1980)
32. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)
33. Wuensche, A.: Self-reproduction by glider collisions: the beehive rule. *International Journal Pollack et al*, pp. 286–291 (2004)

Towards Patterns of Comfort: A Multilayered Model Based on Situated Multi-agent Systems

Paola Lembo, Lorenza Manenti, and Sara Manzoni

CSAI - Complex Systems & Artificial Intelligence Research Center
Department of Informatics, Systems and Communication
University of Milano-Bicocca

{lembo,manenti,manzoni}@disco.unimib.it

<http://www.csai.disco.unimib.it>

Abstract. The paper presents the agent-based model we developed to study crowd dynamics in multi-cultural aggregation contexts. Social and cultural aspects (in particular derived from proxemics theory) are explicitly modeled in order to study the social network resulting from local spatial interactions and cultural differences. To this aim, an agent-based model based on SCA*PED (Situated Cellular Agents for PEdestrian Dynamics) is presented, where pedestrian dynamics result from the local interaction and behavior of an heterogeneous system of autonomous entities situated into a structured environment. The proposed model represents pedestrians' behaving according to local information and knowledge on two separated yet interconnected layers representing different aspects of the overall system dynamics (i.e. Spatial and Proxemic layer). The model explicitly represents on Proxemic layer how cultural differences can influence the perception of neighbors. The model is presented as a formal approach to study comfort properties in spaces where multicultural crowds share a limited structured environment.

1 Introduction

In this paper we describe partial results of a multidisciplinary research we are conducting in order to develop a modeling and computational model for heterogeneous crowd systems in which cultural differences of crowd members are explicitly considered. Social and cultural aspects are explicitly represented in the agent-based model in order to take into account heterogeneity in the system of pedestrians who behave locally (i.e. according to local information) and interact at a physical level (i.e. due to limited shared space).

Traditional modeling approaches (mainly in computer science, fire engineering, building and urban design and planning) focus on pedestrian dynamics with the aim of supporting decision-makers and managers of crowded spaces and events [1] [2] [3]. However, some multidisciplinary proposals have recently been suggested to tackle the complexity of crowd dynamics by taking into account emotional, cultural and social interaction concepts [4] [5] [6] as well.

In this paper we propose an extension of an agent-based approach previously presented to study pedestrian dynamics (i.e. SCA*PED, Situated Cellular

Agents for PEdestrian Dynamics [7]) towards a multi-layered model. According to agent-based modeling and simulation, crowds are studied as complex systems whose dynamics result from local behavior among individuals and their interactions with their surrounding environment [3] [8].

After an outline of the main concepts of Hall's theory [9] [10] on perceived distance and proxemic behavior, we will describe the model based on SCA*PED modeling approach in which proxemic behavior and perceived distance concepts are included. SCA*PED models pedestrian dynamics as resulting from the local interaction and behavior of an heterogeneous system of autonomous entities. The proposed multi-layered model represents pedestrians behaving according to local information and knowledge on two separated yet interconnected layers representing different aspects of the overall system dynamics (i.e. *Spatial* and *Proxemic* layer). The Proxemic layer explicitly models heterogeneities in system members from the view point of the perception of neighboring individuals due to cultural differences. Whenever a local spatial interaction occurs at Spatial layer, the involved entities react differently according to their cultural specifications. In this paper we represent cultural differences according to Hall's theory. Such differences imply different perceptions which on their side allow the study of dynamic comfort properties given a multicultural crowd sharing a structured environment.

2 Perceived Distance and Proxemic Behavior

Proxemic behavior includes different aspects which could it be useful and interesting to integrate in crowd and pedestrian dynamics simulation. In particular, the most significant of these aspects being the existence of two kinds of distance: *physical* distance and *perceived* distance. While the first depends on physical position associated to each person, the latter depends on proxemic behavior based on culture and social rules. The term *proxemics* was first introduced by Hall with respect to the study of set of measurable distances between people as they interact [9]. In his studies, Hall carried out analysis of different situations in order to recognize behavioral patterns. These patterns are based on people's culture as they appear at different levels of awareness.

In [10] Hall proposed a system for the notation of proxemic behavior in order to collect data and information on people sharing a common space. Hall defined proxemic behavior and four types of perceived distances: *intimate distance* for embracing, touching or whispering; *personal distance* for interactions among good friends or family members; *social distance* for interactions among acquaintances; *public distance* used for public speaking. Perceived distances depend on some elements which characterized relationships and interactions between people: posture and sex identifiers, sociofugal-sociopetal¹ (SFP) axis, kinesthetic factor, touching code, visual code, thermal code, olfactory code and voice loudness.

¹ These terms were first introduced in 1957 by H. Osmond in [11].

3 The Two-Layered MAS

In order to integrate some aspects of proxemic behavior into an agent-based model of a crowd, we defined a constellation of interacting *Multi Agent Systems* (MAS) situated on a two-layered structure (i.e. *Spatial* and *Proxemic* layers in Figure 1). Following the SCA*PED approach definition, in each structure, the agents are defined as reactive agents that, as effect of the perception of environmental signals and local interactions with neighboring agents, can change their internal state or their position on the environment. According to SCA framework, each layer is defined by a triple

$$\langle Space, F, A \rangle$$

where *Space* models the environment in which the set *A* of agents is situated, acts autonomously and interacts through the propagation/perception of the set *F* of fields. The *Space* is modeled as an undirected graph of sites (i.e. $p \in P$). Each $p \in P$ is defined by $\langle a_p, F_p, P_p \rangle$, where $a_p \in A \cup \{\perp\}$ is the agent situated in p , $F_p \subset F$ is the set of fields active in p and $P_p \subset P$ is the set of sites adjacent to p . Fields can be propagated and perceived in the same or different layers. In order to allow this interaction, the model introduces the possibility to export (import) fields from (into) each layer.

In each layer, pedestrians and/or relevant elements of the environment (i.e. active elements) are represented by different types of agents. An agent type $\tau = \langle \Sigma_\tau, Perception_\tau, Action_\tau \rangle$ is defined by:

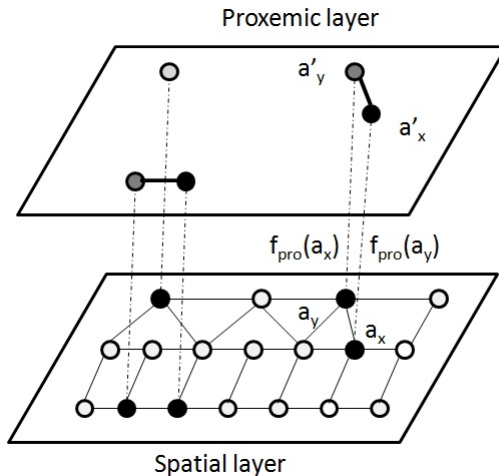


Fig. 1. Two-layered MAS model is shown. Spatial layer describes the environment in which pedestrian simulation is performed while Proxemic layer refers to the dynamic perception of neighboring pedestrians according to proxemic differences. When an agent $a_y \in A_{spa}$ enters the neighborhood of an agent $a_x \in A_{spa}$, both agents emits a field f_{pro} that is perceived by a'_y and a'_x in A_{pro} .

- Σ_τ : the set of states that agents of type τ can assume;
- $Perception_\tau$: a function to describe how an agent is influenced by fields defining a *receptiveness coefficient* and a *sensibility threshold* for each field $f \in F$;
- $Action_\tau$: a function to allow the agent movement between spatial positions, the change of agent state and the emission of fields.

Each agent is defined as a triple $\langle s, p, \tau \rangle$ where τ is the agent type, $s \in \Sigma_\tau$ is the agent state and $p \in P$ is the site in which the agent is situated.

In the remaining of the paper *Spatial* and *Proxemic* layers will be described. The first describing the environment in which pedestrian simulation is performed while the second referring to the dynamic perception of neighboring pedestrians according to proxemic distances.

3.1 The Spatial Layer

In the *Spatial* layer, each spatial agent $a_{spa} \in A_{spa}$ emits and exports to Proxemic layer a field to signal changes on physical distance with respect to other agents. This means that when an agent $a_y \in A_{spa}$ enters the neighborhood of an agent $a_x \in A_{spa}$, both agents emits a field f_{pro} with an intensity id proportional to the spatial distance between a_x and a_y . In particular, a_x starts to emit a field $f_{pro}(a_y)$ with information related to a_y and intensity id_{xy} , and a_y starts to emit a field $f_{pro}(a_x)$ with information related to a_x and intensity id_{yx} . Obviously, $id_{xy} = id_{yx}$ due to the symmetry property of distance and the definition of id .

When physical condition changes and one of the agents exits the neighborhood, the emitting of the fields ends.

Fields are exported into Proxemic layer and influences the relationships and interactions between proxemic agents. How this field is perceived and influences the agent interactions in the Proxemic layer, will be described in the next section.

3.2 The Proxemic Layer

As previously anticipated, *Proxemic* layer describes the agents behavior taking into account some aspects of Hall's theory. Proxemic layer hosts a heterogeneous system of agents where several types of agents τ_1, \dots, τ_n represent different attitudes of a multicultural crowd. Each type τ_i is characterized by a perception function $perc_i$ and a value of social attitude sa_i . This value takes into account all the Hall's categories introduced before and indicates the attitude to sociality for that type of agent.

In this layer, space is described as a set of sites where each site is occupied by a proxemic agent $a_{pro} \in A_{pro}$ and connected to the corresponding site at Spatial layer. Proxemic agents are influenced by fields imported from Spatial layer by means of their perception function. The latter interprets the field f_{pro} perceived, modulating (amplifying or reducing) the value of its intensity id on the basis of sa value associated to agent type.

When in the Spatial layer $a_x \in A_{spa}$ emits a field with information on a_y , in the Proxemic layer $a'_x \in \tau_i$ perceives the field $f_{pro}(a_y)$ as:

$$perc_i(f_{pro}(a_y)) = sa_i \times id_{xy} = ip_{xy} \tag{1}$$

In the same way $a'_y \in \tau_j$ perceives the field $f_{pro}(a_x)$ as:

$$perc_j(f_{pro}(a_x)) = sa_j \times id_{yx} = ip_{yx} \tag{2}$$

Values ip_{xy} and ip_{yx} quantify the different way to perceive the physical distance between a_x and a_y from the point of view of a_x and a_y respectively.

Each $a_{pro} \in A_{pro}$ is also characterized by a state $s \in \Sigma$ which dynamically evolves on the basis of the perceptions of different fields f_{pro} imported from the Spatial layer. The transition of state represents the local change of comfort value for each agent².

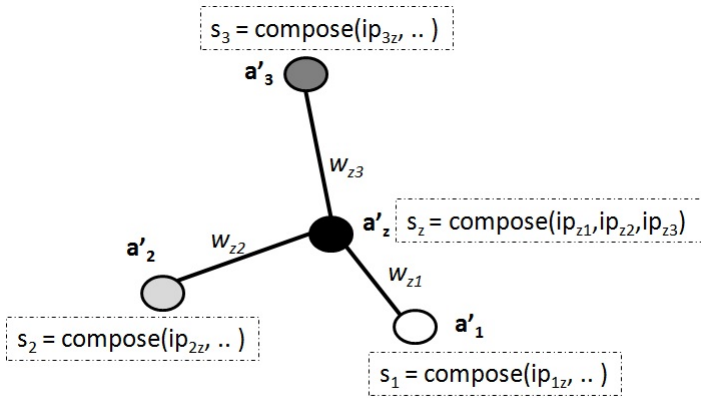


Fig. 2. A system of four agents where the state of agent $a'_z \in A_{pro}$ results from the composition of its perceived neighbors (i.e. $a'_1, a'_2, a'_3 \in A_{pro}$)

In particular, the state evolves according to the composition of the different ip calculated on the basis of interactions which take place in the Spatial layer. Figure 2 shows an heterogeneous system composed of four neighboring agents where the comfort state of each agent results from the composition of all perceived neighbors, that is for each $a'_z \in A_{pro}$:

$$s_z = \text{compose}(ip_{z1}, ip_{z2}, \dots, ip_{zn})$$

where $a'_1, \dots, a'_n \in A_{pro}$ are the corresponding proxemic agents of $a_1, \dots, a_n \in A_{spa}$ which belong to the neighborhood of $a_z \in A_{spa}$.

² State change may imply also a change in the perception. This aspect may be introduced into the model by specifying it into the perception function (i.e. $perc_i(f_{pro}, s) = perc_i(f_{pro})$). Future works will investigate on this issue.

After the perception and modulation of fields perceived, it is possible to consider the relationship between i and j in (1) and (2).

If $i = j$ the two agents belong to the same type (i.e. $\tau_i = \tau_j$) and the values ip_{xy} and ip_{yx} resultant from the perception are equal (i.e. $sa_i = sa_j$ and $id_{xy} = id_{yx}$ for definition). Otherwise, if $i \neq j$ the two agents belong to different types (i.e. $\tau_i \neq \tau_j$) and the values ip_{xy} and ip_{yx} resultant from the perception are different (the agents perceive their common physical distance in different way).

Proxemic relationships among agents are represented as an undirected graph of sites where edges are dynamically modified as effect of spatial interactions (occurring at Spatial layer) and social attitude sa . When a field is perceived from agent a'_x with information on a_y , an edge between a'_x and a'_y is created. When field emission ends (due to the exit of the neighborhood by one agent), the edge previously created is eliminated. The edge (x, y) is characterized by a weight w_{xy} :

$$w_{xy} = |ip_{xy} - ip_{yx}|$$

and represents the proxemic relationships between agents x and y . Obviously, only if $ip_{xy} \neq ip_{yx}$ the w_{xy} is non null.

3.3 Network Evolution on the Proxemic Layer

The evolution of the graph on the Proxemic layer is a dynamical process which depends from spatial changes, so it is possible to study how graph evolves. Let us consider the graph $G = (V, E)$ on Proxemic layer, where V is the set of nodes and E is the set of edges that connect pairs of nodes. In general, the graph G will be composed of areas with connected nodes and areas with non connected nodes. In particular, considering situations in which the density of spatial agents is medium, we can study the properties of the system identifying the heterogeneous and homogeneous areas, their changes and movements. In particular, two interesting cases are:

1. $G = (V, \emptyset)$: the graph is a null graph in which there are no edges. This situation occurs when all proxemic agents belong to the same type τ or when spatial agents are far from each other and there are not fields imported from Spatial layer;
2. $G = (V, E)$ and G is a connected graph (i.e. at least one path connects each couple of agents): this situation occurs when the system is characterized by high heterogeneity at Proxemic layer and high density at Spatial layer.

4 Future Works

This work is part of an ongoing research project with the aim of supporting crowd management of multicultural aggregation contexts, by taking into account cultural attitudes and comfort properties. Studying the dynamics of the Proxemic layer can fruitfully suggest feedback actions on the physical spatial structure that drives pedestrian movement. Preliminary investigations are ongoing about

the study of network properties in order to identify available formal tools for this aim (e.g. ‘small worlds’ networks, the identification of clustering coefficient and the degree distribution [12]).

References

1. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. *Physical Review E* 51(5) (1995)
2. Schadschneider, A., Kirchner, A., Nishinari, K.: CA Approach to Collective Phenomena in Pedestrian Dynamics. In: Bandini, S., Chopard, B., Tomassini, M. (eds.) *ACRI 2002*. LNCS, vol. 2493, pp. 239–248. Springer, Heidelberg (2002)
3. Klugl, F., Rindsfuser, G.: Large-scale agent-based pedestrian simulation. In: Petta, P., Müller, J.P., Klusch, M., Georgeff, M. (eds.) *MATES 2007*. LNCS (LNAI), vol. 4687, pp. 145–156. Springer, Heidelberg (2007)
4. Adamatzky, A.: *Dynamics of CrowdMinds*. Series on Nonlinear Science, vol. 54. World Scientific, Singapore (2005)
5. Was, J.: Multi-agent Frame of Social Distances Model. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 567–570. Springer, Heidelberg (2008)
6. Moussad, M., Garnier, S., Theraulaz, G., Helbing, D.: Collective information processing and pattern formation in swarms, flocks and crowds. *Topics in Cognitive Science* 1(3), 469–497 (2009)
7. Bandini, S., Manzoni, S., Vizzari, G.: Situated Cellular Agents a Model to Simulate Crowding Dynamics. *Special Issues on Cellular Automata E87-D*, 669–676 (2004)
8. Bandini, S., Manzoni, S., Vizzari, G.: Agent Based Modeling and Simulation. In: *Encyclopedia of Complexity and Systems Science 2009*, pp. 184–197 (2009)
9. Hall, T.E.: *The Hidden Dimension*. Doubleday, Garden City (1966)
10. Hall, T.E.: A System for the Notation of Proxemic Behavior. *American Anthropologist, New Series* 65(5), 1003–1026 (1963)
11. Osmond, H.: Function as the basis of psychiatric ward design. *Mental Hospitals* 8, 23–29 (1957)
12. Reka, A., Barabasi, A.: Statistical mechanics of complex networks. *Rev. Modern Physics* 74, 47–97 (2002)

A Pedestrian Movement Model That Takes into Account the Capacity Drop Phenomenon in the Motion of Crowd

Elvezia M. Cepolina* and Alessandro Farina

Department of Civil Engineering, Università di Pisa, Via Diotisalvi 2, 56126 Pisa, Italy

Abstract. In this paper we propose a CA movement model based on the ‘‘Cell Transmission Model’’ developed by Daganzo. The cell transmission model was developed as a discrete approximation to the hydrodynamic theory of vehicular traffic flow. The proposed movement model refers instead to the motion of crowd and is required to simulate the dynamic of an egress process in tall buildings taking into account not only queue and spillback phenomena but also the capacity drop phenomenon. In fact we performed experiments that show important evidence that, for a given cross section, in presence of jam upstream the section, the pedestrian flow through the section is not always equal to the section capacity but suddenly it can drop (dropped capacity). This finding is coherent with recent empirical studies of pedestrian behavior at an exit and in contrast with many previous works where it is assumed that in presence of jam upstream a cross section, the flow through the section equals its capacity. The movement model has been used for simulating evacuation processes in high rise buildings. The target is to assess to which extent the capacity drop phenomenon affects the building evacuation time.

Keywords: Capacity Drop; Movement model; Cellular Automaton; Pedestrian Dynamical Phenomenon.

1 Introduction

The capacity of a cross-section of an escape facility is defined as the maximum of the flow-density function for the given cross-section. It is generally assumed that a jam occurs when the incoming flow exceeds the capacity of a cross-section.

In many previous works, it is assumed that in the presence of a jam upstream of a given cross-section, the flow through the section equals its capacity. In the literature, it is possible to find different specifications to estimate the capacity of a pedestrian facility: Seyfried et al. [1] give a good literature review about this topic. In many studies [2,3,4], capacity is assumed to be a linear function of the cross-section width, in others it is assumed to be a step wise function of the width. In fact, Hoogendoorn and Daamen [5] observed that inside a bottleneck the formation of lanes occurs: this would imply that the capacity increases only when an additional lane can develop.

* Corresponding author.

Tel.: +39 050 2217740; Fax: +39 050 2217762, e-mail: e.cepolina@ing.unipi.it

Recent empirical analysis give evidence that the evacuation time through a door doesn't depend only on geometrical factors, like the door width, but also on the number of conflicts upstream the door. In fact, observations have shown a phenomena called arching, which appears when a big crowd with a high desired velocity tries to pass through a door. In this case the number of conflict is high, the door gets clogged and the crowd gets arch-shaped.

In a laboratory, physicists at the University of Tokyo timed 50 women as they exited as fast as possible through a door 50 cm wide [6]. The average outflow (i.e. the number of evacuees divided by the evacuation time and by the exit width) in case of evacuation in a line was larger than the average outflow in case of a the normal evacuation since there was no conflict. Moreover the data show that the evacuation time decreases if an obstacle is put in front of the exit in a proper way. They got evidence that the pedestrian average outflow increases since the obstacle decreases conflicts at the exit by blocking the pedestrian movements. Therefore the obstacle reduces the inter-pedestrian pressure in front of the door and decreases the magnitude of clogging. They also discovered that the average outflow depends on the position of the obstacle.

Microscopic models have the advantage to take into account pedestrian flow as a collective motion of individuals and are able to represent important features of flow dynamic, like location of moving queues in the network, spillbacks and their dissipation. According to these approaches, the flow patterns result from the impulsive reactions of each single pedestrian to other pedestrians or to the environment within his local surrounding.

Microscopic models are able to take into account conflicts between pedestrians and thus the arching phenomenon from which the average outflow depends are the Social Force model by Helbing and the Floor Field Model by Schadschneider et alii. The Social Force model is continuous in space and uses Newton's equation; to specify the behavior of people in panic, Helbing includes a Heavyside-function, which starts to contribute as soon as the forces or the density get to high [7]. The Floor Field model uses Cellular Automata and therefore is discrete in space; it takes into account conflicts by the frictional function which is a function of the number of pedestrians involved in the conflict [6].

These models are a useful support in the design of the environment details since they are able to assess the effects on the evacuation process of obstacle size and position and of emergency exit widths.

We studied pedestrian flow against time through a bottleneck.. Our experiment outcomes confirm that, given a bottleneck, the average outflow changes a lot as a function of pedestrian speed and accumulation upstream the bottleneck. Moreover the experiment results provide evidence that, in the presence of a jam upstream of the bottleneck, the flow through the bottleneck section is not always equal to the maximum capacity but it can suddenly drop (dropped capacity).

The fact that a bottleneck capacity might change in time and specifically can drop when high density conditions happen upstream should be taken into account in pedestrian evacuation.

The proposed movement model refers to the motion of crowd and is required to simulate the dynamic of an egress process in tall buildings taking into account not only queue and spillback phenomena but also the capacity drop phenomenon. A first

release of the model was presented in [8]. Afterwards the model has been improved and the capacity drop phenomenon has been introduced [9].

The paper is structured in the following way. Section 1 concerns the capacity drop phenomenon in pedestrian flows. Outcomes from empirical studies of the phenomenon and of its activation conditions are reported in this section. In section 2 the proposed movement model is described and in section 3 its application to a case of study is presented. Conclusions follow.

2 The Capacity Drop Phenomenon in Pedestrian Flows

For studying deeply pedestrian behaviour in oversaturated conditions experiments were organised within the SPIRAL project founded by the Royal Society. The experiments took place in the PAMELA laboratory which is a controlled environment at the University College London [10].

We set a corridor with a fixed width of 140 cm followed by a restriction: this change in geometry acts as a bottleneck. The restriction width was 90 cm.

47 participants came to take part in the experiments: the majority of them were students. In each experiment the participants were asked to walk at a predefined speed through the corridor and the restriction.

Since there was only one bottleneck in the experiment layout, congestion occurred only at the bottleneck entrance and not downstream.

Pedestrians were asked to walk at different speeds: at their normal speed (NS), as quick as possible (AQAP) and then we tried to force the pedestrian flow: some ‘spies’ were placed among the pedestrians to force the speed a little (F). We repeated each experiment twice and therefore we collected data on 6 experiments (1 restriction width, 3 speeds and 2 runs for a given width and a given speed).

Pedestrian flow was videotaped from above at three cross sections: the InFS cross section, held at 3.6m before the restriction, at the beginning of the bottleneck and at the OutFS cross section, held at 2m down the bottleneck. The data were analysed manually.

Figure 1 refers to the passage of 45 students walking as quick as possible through the 90 cm width restriction. The cumulative number of pedestrians through the OutFS cross section is plotted against time. The line is the regression on the experimental data: it has two inclinations meaning that the flow through the cross section assumes two values: a higher one (1.9 pedestrians per second) first and a lower one (1.5 pedestrians per second) latter. Since there was congestion upstream the bottleneck when we registered these two values for the outflow, we call the maximum value capacity and the lower one dropped capacity.

In almost all the experiments with the 90 cm wide bottleneck, the flow through the bottleneck equals first the bottleneck capacity and then, when pedestrian accumulation increases upstream, the capacity drops.

In figure 1 the flow breakdown is about 23% and takes place at second 271.

In table 1 are summarised the results of the experiments. The rows refer to the experiment runs (for instance, NS1 is the first run of the experiment where pedestrians walked at normal speed). N_f and N_l are the orders of the first and of the last pedestrian that passed the section at a constant flow rate. T_f e T_l are the evacuation times of the first and last pedestrian respectively. We call capacity the higher constant flow

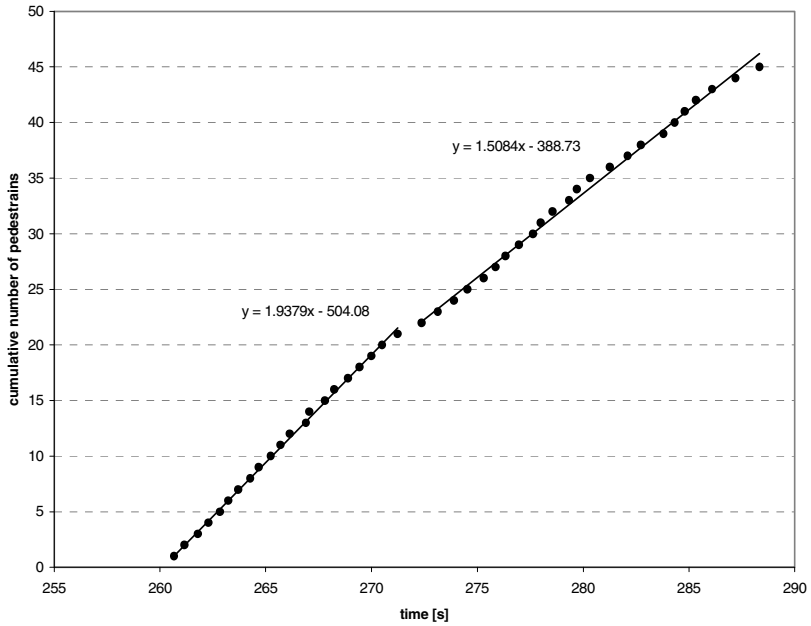


Fig. 1. Result of the experiments held at PAMELA laboratory

Table 1. Results of the experiments

BN 90cm					Capac- ity (ped/s)				Dropped capacity (ped/s)		Average capacity (ped/s)		Outflow (ped/s)
	Nf	Tf	NI	TI	Nf	Tf	NI	TI					
NS1	1	67.7	23.0	81.7	1.6	23.0	81.7	47.0	103.3	1.1	1.3		
NS2	1	158.4						47.0	194.6		1.3	1.3	
AQAP1	1	260.7	21.0	271.2	1.9	21.0	271.2	44.0	287.2	1.4	1.6		
AQAP2	1	344.8	25.0	358.4	1.8	26.0	359.0	46.0	374.2	1.3	1.5	1.6	
F1	1	429.0	27.0	441.5	2.1	27.0	441.5	45.0	453.6	1.5	1.8		
F2	1	520.6	18.0	529.4	1.9	18.0	529.4	45.0	547.4	1.5	1.6	1.7	

rate, and dropped capacity the lower one. Average capacity refers to the time required by all the pedestrians to cross the bottleneck in the given experiment run. Outflow is the average of the “average capacities” related to the two runs of the same experiment.

The table shows that the outflow can assume quite different values and it depends a lot from pedestrian speed. The outflow increases when pedestrian speed increases. We had evidence of capacity drop in 5 over 6 experiments: it ranged from 18 % to 28.5%.

It seems that, for a given geometry, capacity drop is higher when pedestrian speed is higher. The mechanism of these drops was initiated when pedestrian accumulations near the restriction became too high (above a critical value). Having an accumulation above the critical value resulted a necessary but not sufficient condition for a drop. In fact the data from the experiments confirm that capacity drop is a stochastic event in pedestrian flow.

3 The Movement Model

The movement model is based on the “Cell Transmission Model” developed by Carlos Daganzo [11]. The movement model is discretized in space and time.

The time steps have equal duration and the space discretization is strictly linked to the chosen time step.

The building environment is modeled as a cellular automata network.

The corridors/stairways have been divided into homogeneous sections (cells) whose lengths equal the distance travelled by free-flowing traffic in one time step. Thus cells with the same length are cells on which the maximum individual free flow speed is the same. Stairwell cells are shorter than the corridor cells because individual speed downstairs is lower than the one obtained on a flat surface.

The building we are able to take into account neglects flow diverging: the model is able to reproduce pedestrian flow through a sequence of cells or the merging of flows from two different cells.

When evacuation starts, all the building population is assumed to be in the rooms. Each room is a source of pedestrians. When the simulation starts, all pedestrians in the same room start to travel to the exit. The movement model does not take into account pre-movement times.

The state of each cell is defined by the following parameters:

$n_i(t)$ = number of individuals contained in cell i at the beginning of time step t ;

$y_i(t)$ = inflow to cell i during time step t ;

$Q_i(t)$ = maximum number of individuals that can flow into cell i during time step t ;

N_i = maximum number of individuals that can be present on cell i .

N_i^* = critical value of pedestrian accumulation on cell i : when pedestrian accumulation on cell i reaches this value, capacity drop occurs on the downstream cell.

Q_i = maximum capacity of cell i ;

k = capacity drop entity, where $k \in [0;1]$

The maximum number of individuals that can flow into cell i during time step t can assume two values, depending on the circumstances:

$$Q_i(t) = \begin{cases} Q_i & \text{if } n_{i-1}(t) \leq N_{i-1}^* \\ (1-k)Q_i & \text{if } n_{i-1}(t) > N_{i-1}^* \end{cases} \quad (2.1)$$

When the accumulation on the upstream cell is below or equal to the critical value, the maximum number of individuals that can flow into cell i during time step t is given by the maximum cell capacity. When the accumulation on the upstream cell is above the critical value, the maximum number of individuals that can flow into cell i during time step t is lower since capacity drop takes place. We assume an average value for the drop and a deterministic activation condition.

A new generation is created (advancing t by 1), according to the following fixed rule that determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood.

In case of a sequence of cells: $(i-1, i, i+1)$, the cell's neighborhood includes the previous cell and the following one, and the flow propagation is described by the following equations (2.2):

$$\begin{aligned} n_i(t+1) &= n_i(t) + y_i(t) - y_{i+1}(t) \\ y_i(t) &= \min \{ n_{i-1}(t), Q_i(t), N_i - n_i(t) \} \end{aligned} \tag{2.2}$$

The first equation updates the cell state at the beginning of time step $t+1$. The updating refers only to the previous time step. The number $n_i(t+1)$ of individuals on cell i at the beginning of time step $t+1$ is a function of:

- how many individuals were on the cell at the beginning of the previous time step: $n_i(t)$,
- how many individuals entered the cell during the previous time step: $y_i(t)$,
- how many individuals left the cell during the previous time step. This number equals, for flow conservation, the number of individuals that entered the adjacent downstream cell during the same time step: $y_{i+1}(t)$.

The second equation determines the number $y_i(t)$ of individuals that can enter cell i during time step t .

The flow $y_i(t)$ that can enter cell i is the minimum between the demand that would like to enter the cell and the cell constraint. The demand is given by the maximum flow that can be sent by the upstream cell $n_{i-1}(t)$. The cell constraint is determined by the minimum between:

- the maximum number of individuals that can flow into cell i during time step t , $Q_i(t)$;
- the space availability on the cell, which is given by the difference between the maximum number of individuals that could be contemporaneously present on the cell,
- and the current number of individuals on the cell, $N_i - n_i(t)$.

In case of merging cells, we address the reader to Cepolina [8].

The movement model has been implemented in an object-oriented simulator. Cells are objects and are seen as being 'intelligent' – albeit static - agents which are able to move individuals forward according to the movement model previously described. Cells' internal state is described by $n_i(t)$, $y_i(t)$ and $Q_i(t)$.

4 Case Study

The model has been applied to calculate the evacuation time (i.e. the time required for emptying a building) from high-rise buildings. The evacuation time is firstly calculated according to the capacity drop phenomenon, then neglecting this phenomenon.

The time step is assumed equal to 1s.

Each floor of the building has an unique central corridor about 17 m long and 1,4 m wide. The corridor cells are 1.2 m long, because the maximum pedestrians' speed is

assumed equal to 1.2 m/s. The central corridor is therefore represented by a sequence of 14 cells. Each floor has 4 rooms, which are connected directly to the corridor and each room has a load of 20 pedestrians.

Floors are connected together through two stairs: each end of the corridor in the floor is connected to a stairwell. The two stairwells are near the emergency exits, which are located at each end of the ground floor's corridor. Each stair is long 9m and wide 1.4m. Because the maximum speed on stairs is assumed equal to 0.9 m/s, the stairway cells are 0.9 m long.

The egress paths are fixed: at each floor people in two rooms on the right hand side exit the building through the right stairwell and emergency exit 1, people in the two rooms on the left hand side exit through the left stairwell and emergency exit 2. Thus, all the cells can be crossed only by unidirectional flow.

Room doors and exit doors have a width of 0.9 m, therefore they represent bottlenecks. At bottlenecks, and at the merging sections, in which critical density can be exceeded, capacity drop may occur.

Five different scenarios have been considered, i.e. the building having one (i.e. the ground floor), two, three, four and five floors. The scenarios in exam are represented in figures 2. A detailed representation of corridors and stairs is given in figure 2 only for the ground floor.

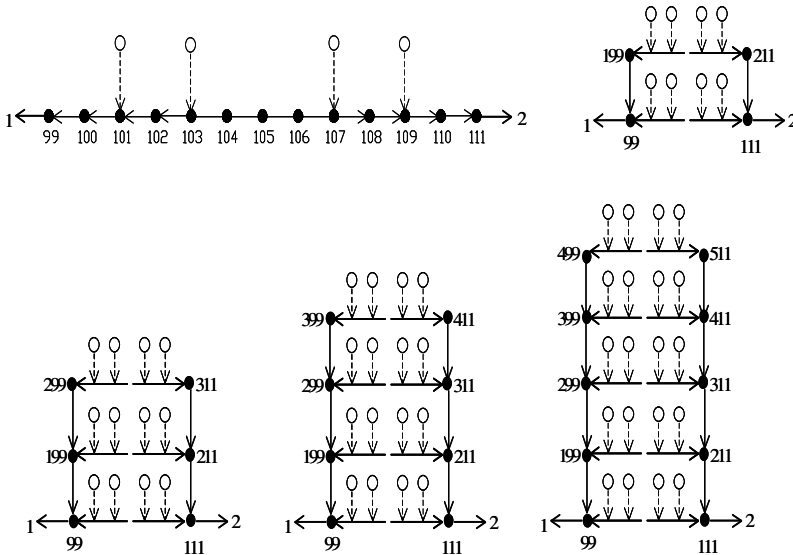


Fig. 2. The simulated scenarios

The model has been calibrated through data collected in the experiments carried out in the PAMELA laboratory.

Cell characteristics are reported in Table 2. We assume $k = 0.27$ since the capacity drop which occurs in the case of forced flow is equal to 27%.

Table 2. Cell characteristics; τ is the time step

	length (m)	width (m)	Q_i (n° ped / τ)	N_i (n° ped)	N^*i (n° ped)	k
Room/exit		0.9	3	50	3	0.27
Corridor cells	1.2	1.4	4	4	4	0.27
Stairway cells	0.9	1.4	4	3	3	0.27

During the simulation activity we incurred in the following problem due to time discretisation. Since k is a real number; $(1-k)Q_i$ is a real number. As it represents the possible number of people able to enter a section in a time step, it has to be an integer. If the decimal portion of $(1-k)Q_i$ is 0.5 or greater, the return value is equal to the smallest integer greater than $(1-k)Q_i$. Otherwise, the simulator returns the largest integer less than or equal to $(1-k)Q_i$. This has two kind of consequences: first, if $(1-k)Q_i$ results lower than 0.5 because Q_i is small or the capacity drop is high, the simulator does not allow anybody to enter cell i until the capacity drop is active there. Secondly, if Q_i is small, for instance 2 pedestrians per second, the simulation results show that a capacity drop of 50% has the same effects as a drop of 30%. These drawbacks could be avoided if Q_i assumes proper values: this could be achieved with a proper time step selection.

In figure 3, evacuation times, expressed in seconds, are reported, and their values are obtained both under the hypothesis of the capacity drop phenomenon and without this assumption.

The simulation results show that obviously the evacuation time is strongly dependent on the number of floors in the building (and therefore the building population). Moreover, capacity drop occurrences affect heavily the results, and neglect the capacity drop phenomenon leads to a great underestimation of evacuation times. This underestimation increases with the number of floors of the building (and therefore with the building population), as shown in figure 3, in particular when the number of floors goes from 3 to 4.

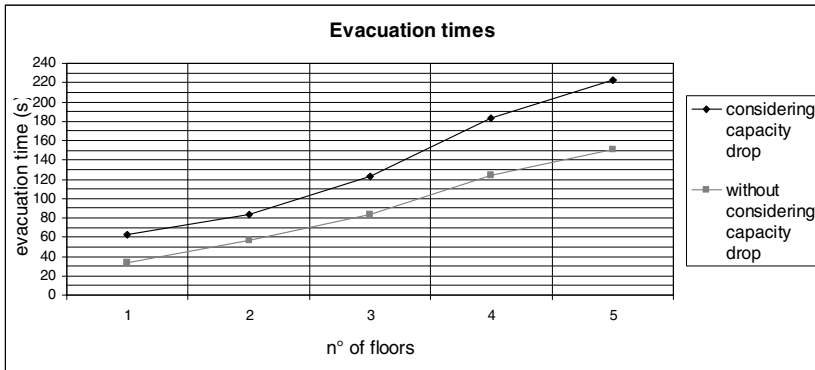


Fig. 3. Evacuation time referred to number of floors

5 Conclusions

The movement model discussed in the paper refers to the motion of crowd and is required to simulate the dynamic of an egress process in tall buildings taking into account not only queue and spillback phenomena but also the capacity drop phenomenon.

With the target of understanding and quantifying this phenomenon empirical studies have been performed. The data have been used to calibrate the movement model. However, the statistical distribution of capacity drops is not known because enough empirical data have not yet been collected. Thus, the accuracy of the results has not yet been examined.

The cellular automata approach resulted convenient since it is easy adaptable to different buildings and since it is frugal on CPU power and thus can handle greater amounts of pedestrians. This last aspect is very important since the target of the movement model is to reproduce the dynamic of an egress process in high rise buildings.

References

1. Seyfried, A., et al.: New insights into pedestrian flow through bottlenecks (2007), <http://arxiv.org/abs/physics/0702004v1>
2. Predtechenskii, V.M., Milinskij, A.I.: Planning for Foot Traffic Flow in Buildings. Stroiizdat Publishers, Moscow (1969); Translated and published for the National Bureau of Standards by Amerind Publishing Co. Pvt. Ltd., New Delhi (1978)
3. Weidmann, U.: Transporttechnik der Fußgänger. Schriftenreihe des IVT, vol. 90. ETH, Zurich (1993)
4. Nelsen, H.E., Mowrer, F.W.: Emergency movement. In: DiNenno, P.J. (ed.) SFPE Handbook of Fire Protection Engineering, 3rd edn. National Fire Protection Association, Quincy (2002)
5. Hoogendoorn, S.P., Daamen, W.: Pedestrian Behaviour at Bottlenecks. CODEN: TRSCBJ 39(2), 147–159 (2005), Print ISSN: 0041-1655, Electronic ISSN: 1526-5447
6. Yanagisawa, D., et al.: Introduction of Frictional and Turning Function for Pedestrian Outflow with an Obstacle. physics.soc-ph (2009)
7. Helbing, D., Farkas, I.J., Molnar, P., Vicsek, T.: Simulation of Pedestrian Crowds in Normal and Evacuation Situations. In: Schreckenberg, M., Sharma, S.D. (eds.) Pedestrian and Evacuation Dynamics, pp. 21–58. Springer, Berlin (2002)
8. Cepolina, E.: A methodology for defining building escape routes. Civil Engineering and Environmental Systems 22(1), 29–47 (2005)
9. Cepolina, E.: Phased Evacuation: An optimisation model which takes into account the capacity drop phenomenon in pedestrian flows. Fire Safety Journal 44, 532–544 (2009)
10. Cepolina, E., Tyler, N.: Understanding Capacity Drop for designing pedestrian environments. In: 6th international Walk21 Conference, Zurich, September 22-23 (2005)
11. Daganzo, C.: The cell transmission model: network traffic. Transportation Research part B, 79–93 (1995)

A Cellular Automaton Model for Crowd Evacuation and Its Auto-Defined Obstacle Avoidance Attribute

Ioakeim G. Georgoudas, Georgios Koltsidas, Georgios Ch. Sirakoulis,
and Ioannis Th. Andreadis

Democritus University of Thrace, Department of Electrical and Computer Engineering,
Laboratory of Electronics, GR 67100 Xanthi, Greece
{igeorg, georkolt, gsirak, iandread}@ee.duth.gr

Abstract. In this paper, a crowd evacuation model based on Cellular Automata (CA) is described. The model takes advantage of the inherent ability of CA to represent sufficiently phenomena of arbitrary complexity and to be simulated precisely by digital computers as well. Pedestrian movement depends on their distance from the closest exit, which is defined dynamically. The adoption of Manhattan distance as the reference metric provides calculation simplicity, computational speed and improves significantly computational performance. Moreover, the model applies an efficient method to overcome obstacles. The latter is based on the generation of a virtual field along obstacles. A pedestrian moves along the axis of the obstacle towards the direction that the field increases its values, leading her/him to avoid the obstacle effectively. Distinct features of crowd dynamics and measurements on different distributions of pedestrians have been used to evaluate the response of the model.

Keywords: Cellular Automata, Crowd Modeling, Pedestrian Evacuation, Obstacle Avoidance.

1 Introduction

Various models that try to efficiently approach crowd movement during evacuation have been presented in literature, proposing theoretical and applicable solutions. Deep insight in crowd dynamics has resulted in better understanding of pedestrian behaviour as well in substantial changes regarding the architecture of such constructions. Crowd safety and comfort in highly congested places not only depend on the design and the function of the area, but also on the behaviour of each individual. Results prove that people under panic tend to lose their individuality, display herding behaviour and it is possible not to make use of means of emergent evacuation effectively [1].

Pedestrian dynamics have been reported following a great variety of approaching methods, thus indicating the importance of the issue. In particular, CA-inspired methods as well as lattice-gas and social force models or agent-based and fluid-dynamic methods have been proposed to investigate and reveal the attributes of crowd evacuation [2]. All approaches can be qualitatively distinguished, focusing on different characteristics that each of them dominantly display.

On the other hand, evacuation could be defined as a non linear problem with numerous factors affecting it. A system of partial differential equations could effectively approach it, but this would lead to a very computationally demanding system, in terms of processing time, complexity as well as power consumption. CA can certainly act as an alternative to such systems, because they can compute values of physical quantities over finite areas (CA cells) at discrete time steps. Literature reports various CA-based models investigating crowd behaviour under various circumstances. Interactions among pedestrians, friction effects [3] and herding behaviour [4] as well as the impact of environmental conditions [5] and bi-directional pedestrian behaviour [6] has been examined. Furthermore, CA models that focus on human behaviours, such as inertial effects, unadventurous effect and group effect have been also developed [7].

2 The CA Model and Characteristic Measurements

The presented model is based on CA; hence its simulation mechanism is matrix-driven, discretising a floor area into a grid. The grid of the automaton is homogeneous and isotropic, while the CA cells are able to exist in two possible states; either free or occupied by exactly one particle. Each cell is equivalent to the minimum area, which a person could occupy and defined equal to $0.4m \times 0.4m$ [8]. During each time step, an individual chooses to move in one of the eight possible directions of its closest neighbourhood. Each particle moves towards the direction which is closest to an exit.

The motion mechanism issues from a potential field approach, based on Manhattan metric, which is calculated by the following equation:

$$\left\| \vec{x} \right\| = |\Delta x| + |\Delta y| \quad (1)$$

The gradient descent on the potential function defines the direction of movement, thus introducing a kind of global space knowledge for pedestrians. The model is inherently emergent, as the interactions among simple parts can simulate complex phenomena such as crowd dynamics.

The method used for the calculation of the potential field is a flood fill method, where the distance is calculated by moving a cell to closest neighbour cell and summing up the distances [9]. This operation is recursively applied to all surrounding cells. If a cell is occupied by an obstacle then this obstacle cell will not flood its neighbours. Algorithmically, the aforementioned method is based on an $n \times 9$ matrix, calculated for each occupied cell. The elements represent all possible updated spatial and temporal states of the occupied cell (Fig. 1). Variable n indicates the number of the exits. Each element of the i -th row ($i=1, \dots, 9$) specifies the distance of the occupied cell and its eight neighbours from the i -th exit. The occupied cell is always represented by the fifth cell of each row, whereas all other cells represent the eight closest neighbours, i.e. the north-west (NW), north (N), north-east (NE), west (W), east (E), south-west (SW), south (S) and south-east (SE) neighbour, respectively. As soon as all possible routes are detected, i.e. as soon as each of the $n \times 9$ elements of the matrix is calculated, the shortest prevails. Consequently, both the destination exit and the direction during next step are defined. The former is represented by the row of the minimum value element and the latter is indicated by its column.

$i=1$	NW	N	NE	W	cell of reference	E	SW	S	SE
.
.
.
$i=n$	NW	N	NE	W	cell of reference	E	SW	S	SE

Fig. 1. The $n \times 9$ matrix

Fig. 2 further clarifies the operation of the matrix, assuming the special case of a unique exit ($n=1$). Then, the size of the matrix becomes 1×9 and its elements can be rearranged in a 3×3 form. The latter form illuminates the value of each element. It corresponds to the distance of each cell, i.e. the occupied cell and its closest neighbours, from the exit. Distance definition originates from Manhattan metric; the minimum number of steps, in order a pedestrian to reach the exit moving strictly vertically or horizontally is calculated. More details regarding the structure and application of the Manhattan distance to the CA crowd model can be found in [10].

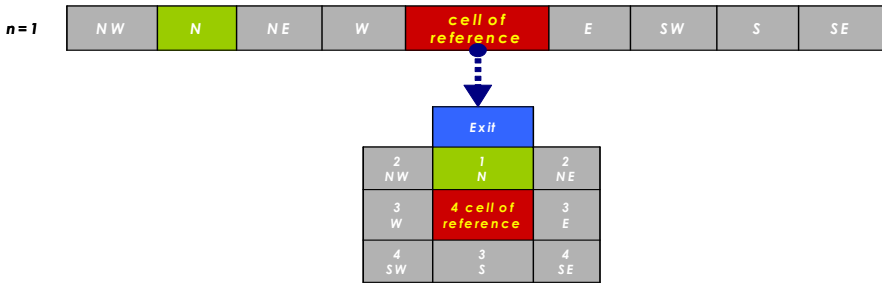


Fig. 2. The special case of unique exit; $n=1$ and the corresponding rearrangement of the $n \times 9$ matrix to a 3×3 form.

3 The Auto-Defined Obstacle Field Effect

A distinct feature of the model is an automated process that enables obstacle avoidance based on the effect of a virtual field generated near obstacles. Inside the field, a pedestrian moves towards the direction of greater field values. Following that direction a pedestrian is enabled to overcome efficiently even complex obstacles. Specifically, in the general case, obstacle field values are increasing forming a parabola, which is described by the following equation:

$$F(x) = \frac{1}{2p}(x - x_o)^2, p > 0 \tag{2}$$

$$\forall \in \{[x_{wl}, x_{wr}] \cap (x_A - x_{wl} \neq 0 \cup x_{wr} - x_B \neq 0)\}$$

$$x_o = \frac{x_B - x_A}{2} \quad (3)$$

In equation (2), p corresponds to the parameter of the parabola, which also defines the distance between the two branches of the graphical representation of the function. In fact, as $\frac{1}{2p} \rightarrow 0$, then the width of the parabola increases. Moreover, (x_A, y_A) , (x_B, y_B) represent the coordinates of the edges of the obstacle, whereas, x_{wl} , x_{wr} correspond respectively to the very left and very right x -axis coordinate of the walls. Equation (3) defines x_o , which corresponds to the x -axis coordinate of the middle point of the obstacle.

In case that the obstacle is bonded to a wall, then the field is generated according to the common coordinate of the obstacle and the wall, as described by equations (4) and (5):

$$x_A - x_{wl} = 0 \Rightarrow F(x) = \frac{1}{2p}(x - x_{wl})^2 \quad (4)$$

$$x_{wr} - x_B = 0 \Rightarrow F(x) = \frac{1}{2p}(x - x_{wr})^2 \quad (5)$$

The length of the obstacle is given by:

$$L_{obstacle} = x_B - x_A \quad (6)$$

whereas the length of the area between walls is given by:

$$m = x_{wr} - x_{wl} \quad (7)$$

Finally, in the case of complex obstacles the corresponding field is generated by the superposition of fields that correspond to fundamental obstacles. Fig. 3 clarifies the effect of the auto-defined obstacle field to the direction of pedestrian movement, in correspondence to the location and the shape of the obstacle. It should be mentioned that above mathematical presentation takes into account geometrically shaped obstacles, however with slight modification can be successfully applied to arbitrary shaped obstacles as well.

Fig. 4 displays successive snapshots of a simulation example that display the application in software of the aforementioned mathematical method for overcoming complicated obstacles. It is proved the proper application of the method and its qualitative validity as well.

As far as the validity of the model concerns, simulation results prove that distinct attributes of crowd behaviour [11], such as collective effects, blockings in front of exits and random to coherent motion due to a common purpose, transition to incoordination (arching) due to clogging as well as mass behaviour qualitatively characterise the model as well. The model has been further evaluated by means of computer-generated distributions of pedestrians, in order to estimate how well it reproduces flow-density dependence, i.e. to reveal its fundamental diagram response. Specifically, in practice, the global density was measured by counting all pedestrians in the area of interest and then dividing by that area, whereas the global flow was defined by

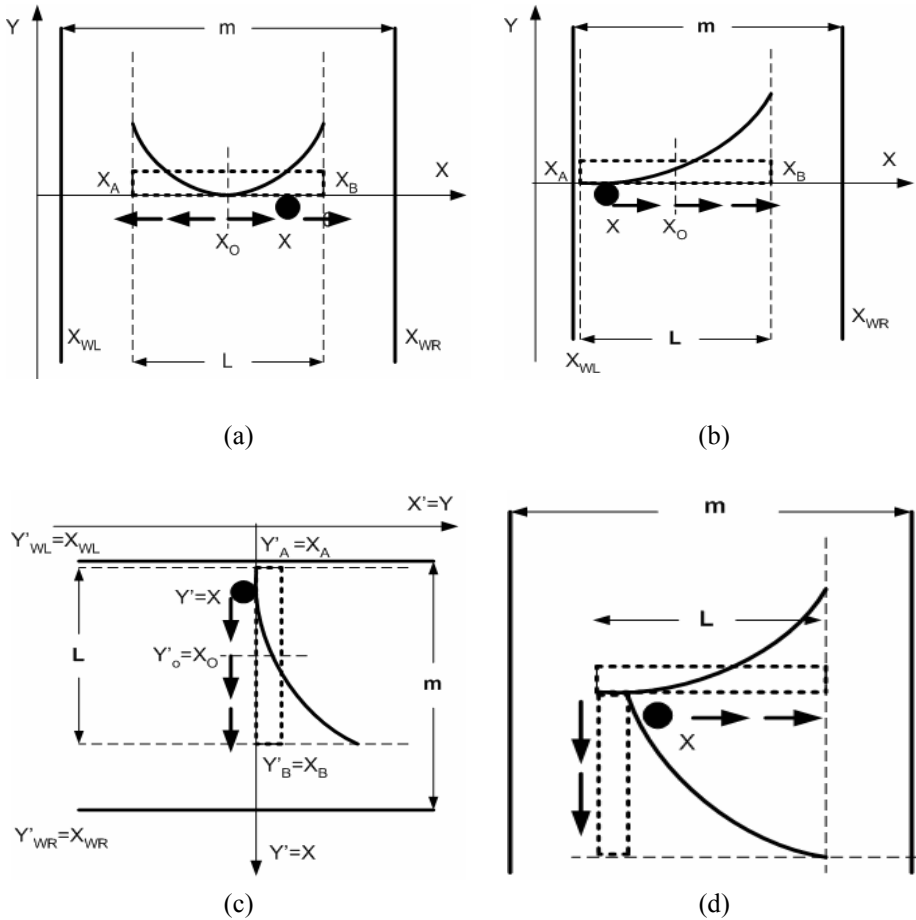


Fig. 3. (a) The graphical representation of the obstacle field in case that the obstacle lays between walls. The values of the field increase in the direction from left to right for half the length of the obstacle and the vice versa for the other half. Thus, the pedestrian is enabled to move following the one direction or the other, as indicated by arrows. (b) The response of the obstacle field, in case that the exit is closer to the left edge of the obstacle ($X_A=X_{wl}$). (c) The case of a vertical obstacle and the corresponding field. (d) The case of a complex obstacle. The final field is generated by the superposition of field cases (b) and (c).

the number of people passing a long cross section per unit time, divided by its length (Fig. 5). The corresponding diagrams quite match with that of A. Johansson et al [12] enhancing the validation of the model. Fig. 6 displays the response of the model in the case of a large number of people walking through a broad corridor with obstacles hampering their movement.

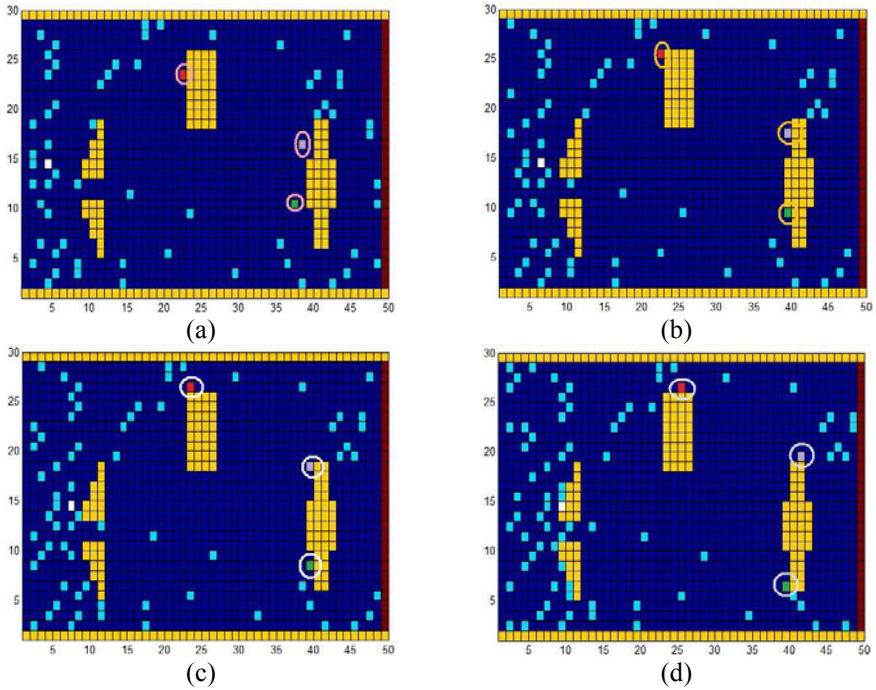


Fig. 4. Successive snapshots of a simulation example that displays the way that pedestrians overcome complicated obstacles based on the auto-defined obstacle field effect.

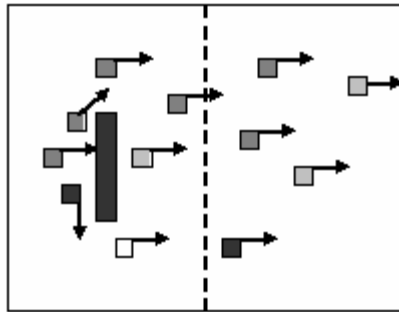


Fig. 5. Global density measurement method. The global density is measured by counting all pedestrians in the area of interest and then dividing by this area.

In order to evaluate in what extent the model is appropriate for quantitative and qualitative of pedestrian flow, its response is compared to empirical velocity-density relations. Assuming that pedestrians move smoothly according to the flow-density relationship derived from fluid-dynamics, the flow per meter width follows the relationship below:

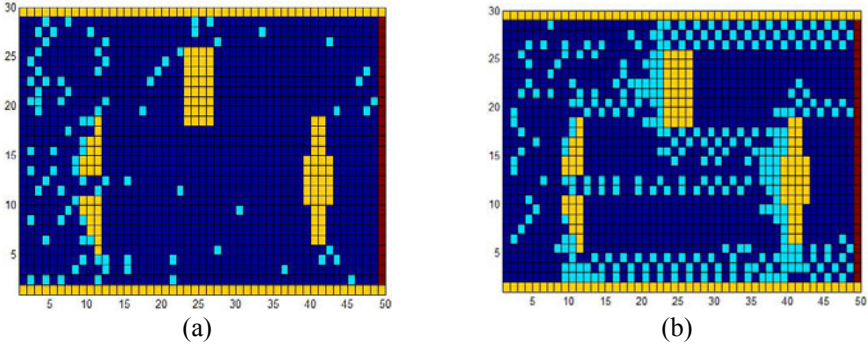


Fig. 6. Snapshots of the simulation example

$$Q(\rho) = \rho V(\rho) \tag{8}$$

where Q represents the flow per meter width and ρ is the pedestrian density [13-14]. Equation (8) is often used for designing pedestrian facilities, for safety and evacuation studies. Empirical measurements are often restricted up to 4-6 *persons/m²* only. In the study [14], for example, the maximum density ρ_{\max} is 5.4 *persons/m²* and the corresponding speed-density relationship is

$$V(\rho) = V_o \left\{ 1 - \exp \left[-a \left(\frac{1}{\rho} - \frac{1}{\rho_{\max}} \right) \right] \right\} \tag{9}$$

where $V_o=1.34$ *m/sec* is the free speed in low densities and $a=1.913$ *persons/m²* a fit parameter. According to the principles of the model, each person can cover a minimum area of $0.4 \times 0.4 = 0.16$ *m²*, hence the maximum density is equal to 6.25 *persons/m²*, which is found in agreement with other literature approaches related to that issue [15-16].

Regarding the simulation example, the scenario adopted is that of a gradually increased number of people walking through a broad corridor. The whole area corresponds to a CA grid of 50×30 *cells* that is equal to an area of 240 *m²*. The area of interest (*AoI*), i.e. the one that takes part in density measurements covers 90 CA *cells*, that is an area of $AoI = 90$ (*cells*) $\times [0.4$ (*m*) $\times 0,4$ (*m*)]/*cell* = 14.4 *m²*. For each of the total 190 times steps of the simulated experiment, the number of people inside the area of interest is measured and the corresponding density is calculated, according to the following relationship:

$$\rho_t = \frac{No(\text{persons})_{\text{insideAoI}}}{AoI} \tag{10}$$

where t corresponds to a specific time step, i.e. $t=1, \dots, 190$.

As soon as all time steps are completed, the maximum density ρ_{\max} (=3.611 *persons/m²* for the specific example) is detected and then the global velocity is computed for each time step, according to equation (9). Finally, the flow per meter width, Q , is evaluated according to equation (8) leading to the graphical representation of the results. Table 1 displays a selection of results derived from simulation.

Table 1. All types of results for randomly selected times steps

Time step	People inside AoI	Total No of people	Density (per-sons/m ²)	Speed (m/sec)	Flow (people/m/sec)
15	2	25	0.139	1.340	0.186
78	41	1450	2.847	0.176	0.500
134	52	3840	3.611	0.007	0.025
185	35	6169	2.431	0.302	0.735

In the following, three characteristic diagrams derived from simulation results are presented. Particularly, the global flow-global density diagram is given in Fig. 7, whereas the speed-density diagram is showed in Fig. 8 and the cumulative flow of pedestrians in Fig. 9. For each curve the corresponding one from A. Johansson et al. [12] is also depicted, in order the response of the model to be compared with existing literature results.

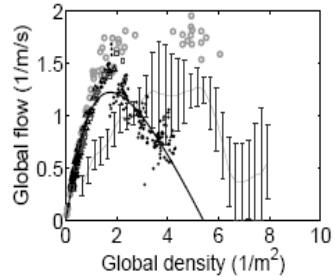
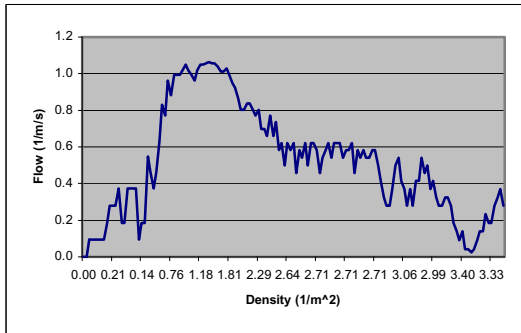


Fig. 7. (a) The global flow-global density diagram as derived by the model. (b) Global flow as a function of the global density in the video-based experiment of A. Johansson et al. [12]. Symbols correspond to the empirical data of Mori and Tsukaguchi [16] (circles), Polus et al. [17] (squares), Fruin et al. [18] (triangles), and Seyfried et al. [13] (dots). The solid fit curve is from Weidmann [14].

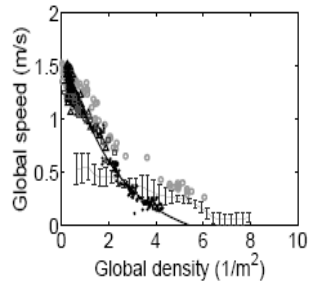
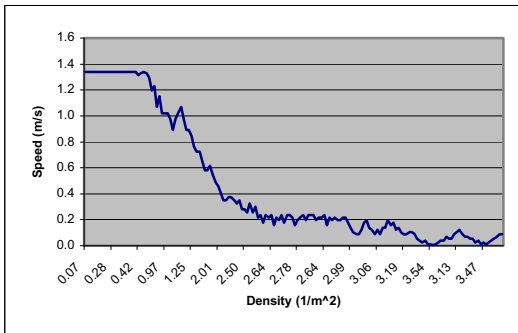


Fig. 8. (a) The speed-density diagram of the simulation example. (b) Global speed as a function of the global density in the video-record based experiment of A. Johansson et al. [12]. Symbols correspond to the empirical data as clarified in the caption of Fig. 7.

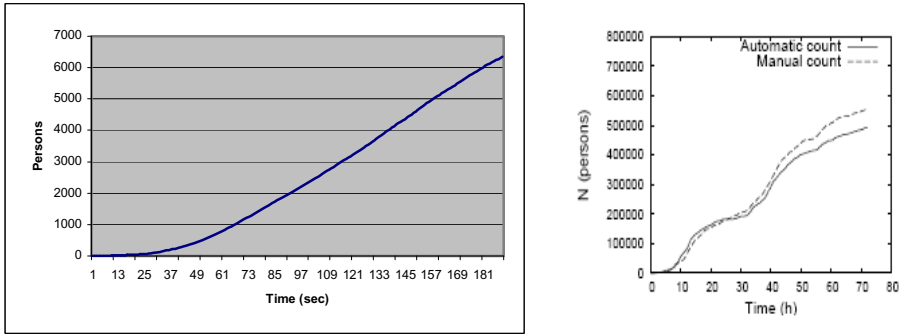


Fig. 9. (a) Cumulative flow of pedestrians in the simulated example. (b) The corresponding diagram in the case of the pilgrims' flow example of A. Johansson et al. [12].

4 Conclusions

A CA model for the simulation of crowd evacuation with auto-defined obstacle avoidance approach has been described. Crowd consists of individuals and macroscopical features of its motion emerge from the local interaction of pedestrians. Motion mechanism stems from a potential field based on the Manhattan distance of each pedestrian from the exits. Each obstacle defines a field around it according to its shape and position. The field affects a pedestrian that reaches it by guiding her/him to move along the axis of the obstacle, towards the direction of increasing field values. Model evaluation includes extended simulation processes that prove the existence of macroscopic characteristic features of crowd dynamics. Pedestrian distribution measurements produce the fundamental relationship between flow and density as well as that of speed-density in good quantitative and qualitative agreement with literature reports [12]. Several different cases with different shaped obstacles and different crowd distributions have been successfully tested. Finally the model is in process of calibration with the use of real data under conditions of increased density, i.e. video-recorder evacuation process of people leaving a football stadium of Greek Super League.

References

1. Goldstone, R.L., Janssen, M.A.: Computational models of collective behaviour. *Trends in Cognitive Sciences* 9(9), 424–430 (2005)
2. Xiaoping, Z., Tingkuan, Z., Mengting, L.: Modeling crowd evacuation of a building based on seven methodological approaches. *Building and Environment* 44, 437–445 (2009)
3. Schultz, M., Lehmann, S., Fricke, H.: A discrete microscopic model for pedestrian dynamics to manage emergency situations in airport terminals. In: Waldau, N., Gattermann, P., Knoflacher, H., Schreckenber, M. (eds.) *Pedestrian and Evacuation Dynamics 2005*, pp. 369–375. Springer, Heidelberg (2007)
4. Nishinari, K., Sugawara, K., Kazama, T., Schadschneider, A., Chowdhury, D.: Modelling of self-driven particles: foraging ants and pedestrians. *Physica A* 372, 132–141 (2006)
5. Yu, Y.F., Song, W.G.: Cellular automaton simulation of pedestrian counter flow considering the surrounding environment. *Physical Review E* 75(046112), 1–8 (2007)

6. Fang, W.F., Yang, L.Z., Fan, W.C.: Simulation of bi-direction pedestrian movement using a cellular automata model. *Physica A* 321, 633–640 (2003)
7. Yuan, W.F., Tan, K.H.: An evacuation model using cellular automata. *Physica A* 384, 549–566 (2007)
8. Burstedde, C., Klauck, K., Schadschneider, A., Zittartz, J.: Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A* 295, 507–525 (2001)
9. Kretz, T., Bönisch, C., Vortisch, P.: Comparison of Various Methods for the Calculation of the Distance Potential Field, <http://arxiv.org/abs/0804.3868>
10. Georgoudas, I.G., Sirakoulis, G.C., Andreadis, I.: Potential Field Approach of a Cellular Automaton Evacuation Model and its FPGA implementation. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) *ACRI 2008*. LNCS, vol. 5191, pp. 546–549. Springer, Heidelberg (2008)
11. Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. *Nature* 407, 487–490 (2000)
12. Johansson, A., Helbing, D., Al-Abideen, Al-Bosta, H.Z.S.: From crowd dynamics to crowd safety: A video-based analysis. *Advances in Complex Systems* 11(4), 497–527 (2008)
13. Seyfried, A., Steffen, B., Klingsch, W., Boltes, M.: The fundamental diagram of pedestrian movement revisited. *J. Stat. Mech.*, P10002 (2005)
14. Weidmann, U.: *Transporttechnik der Fußgänger (Schriftenreihe des Institut für Verkehrsplanung, Transporttechnik, Straßen- und Eisenbahnbau 90, ETH Zurich, Zurich* (1993)
15. Predtechenskii, V.M., Milinskii, A.I.: *Planning for Foot Traffic Flow in Buildings*. Amerind Publishing Co., New Delhi (1978)
16. Mori, M., Tsukaguchi, H.: A new method for evaluation of level of service in pedestrian facilities. *Transportation Research A* 21(3), 223–234 (1987)
17. Polus, A., Schofer, J.L., Ushpiz, A.: Pedestrian flow and level of service. *Journal of Transportation Engineering* 109, 46–56 (1983)
18. Fruin, J.J.: Designing for pedestrians: A level-of-service concept. *Highway Research Record* 355, 1–15 (1971)

A Learning Algorithm for the Simulation of Pedestrian Flow by Cellular Automata

Hideaki Ishii and Shin Morishita

Graduate School of Environment and Information Sciences, Yokohama National Univ.
79-7 Tokiwadai, Hodogaya-ku, Yokohama, 240-8501, Japan
mshin@ynu.ac.jp

Abstract. Cellular Automata was applied to model the pedestrian flow, where the local neighbor and transition rules implemented to each person in the crowd were determined automatically by the experience of pedestrians. The experience was based on two parameters; the number of continuous vacant cells in front of the cell to proceed, and the number of pedestrian in the cell to proceed. The experience was evaluated numerically, and a pedestrian selected the cell to proceed by the evaluated index. The flow formations by pedestrians in the opposite direction on a straight pathway and on a corner were simulated, and the number of rows was discussed in relation to the density of pedestrian on the simulation space.

Keywords: Pedestrian Flow, Learning Algorithm, Local Neighbor Rule, Transition Rule, Density of Pedestrian, Number of Row.

1 Introduction

Cellular Automata (CA) has been considered to be one of the strong tools in modeling complex phenomena such as pattern formation of natural system, multi-phase fluid flow, traffic flow, city logistics or economical activities¹⁻⁶. Corresponding to the application field, CA has been given various names in each field. CA has also been applied to solve partial differential equations. Most of all, CA may be applicable in the simulation of complex systems, where a great number of elements consisting of the phenomena are affected to each other in the system. Pedestrian flow is a typical example of the complex systems; one person walks toward the destination, paying attention to other persons around him or her, and the pedestrian flow may be build up as assemblage of movement of each person.

In the modeling of CA, local neighbor rules and transition rules should be implemented to simulate time evolution of the phenomena to be considered. But these rules have large effect on the simulation results, and careful consideration is required in defining these rules.

In this paper, CA has been applied to the simulation of pedestrian flow, in which the cell for a pedestrian to proceed was determined automatically by experience of each pedestrian. The experience was based on two parameters; the number of continuous vacant cells in front of the cell to proceed, and the number of pedestrian on the cell to

proceed. The experience was evaluated numerically, and a pedestrian selected the cell to proceed by the evaluated index. As examples, the flow formations by pedestrians in the opposite direction on a straight pathway and on a corner were simulated.

2 State Variables and Corresponding Rules

In this chapter, the definition of technical terms used as state variables, the proposed learning algorithm, the local neighbor rules and the transition rules are briefly introduced.

2.1 State Variables

Three types of cell state variables were defined; “pedestrian”, “floor” and “wall” as shown in Fig. 1. The state variable of “pedestrian” indicated the existence of pedestrians in a cell. The movement of pedestrians was simulated by changing the state variable from “pedestrian” to “floor” or from “floor” to “pedestrian” along time progress. In addition, “position coordinate”, “the number of pedestrian” and “guide sign” were defined as state variables. In the simulation, more than one pedestrian were permitted to exist in one cell at the same time. The guide sign represented the information sign in the design space and controlled the movement of pedestrians.

The state variable of pedestrian included additional information; “position”, “goal”, “sight”, “velocity”, “directional degree” and “Learned data”. The goal was the target cell where the pedestrian was heading for. This information was given when a pedestrian was first positioned to the simulation space at the entrance. The state variable of sight was defined as the sight area of a pedestrian as shown in Fig. 2. This information was composed of sight radius and degree. The simulation in this paper defined the sight degree as 0. Velocity was the maximum velocity and the current velocity. The maximum velocity was given when a pedestrian first positioned to the simulation space. Directional degree was defined as movement direction of a pedestrian. This information was composed of the heading and current degree.

The Learned data was composed of the assembly of situation pattern which was composed of “Pattern number” and 5 “Destination cell data”. The Pattern number was a number which indicated the situation of sight area, and expressed by 8 digit-number. This number included the number of same directional pedestrians, the number of opposite directional pedestrians which might cross of the marked pedestrian, the number of opposite directional pedestrians which might not break in course of the marked pedestrian, and the number of wall cells.

The Destination cell data was composed of “candidate cell” to proceed, “experienced value”. The candidate cell was one of five cells shown in Fig. 3. The Experienced value was an indicator of how comfortable the candidate cell was, and was evaluated by the number of vacant cells in front of the candidate cell (α), and the sum of pedestrians in the candidate cells (β). The experienced value (γ) is defined as;

$$\gamma = \frac{\alpha}{\beta + 1}$$

Unit value is added to the denominator, because the minimum of β could be zero.

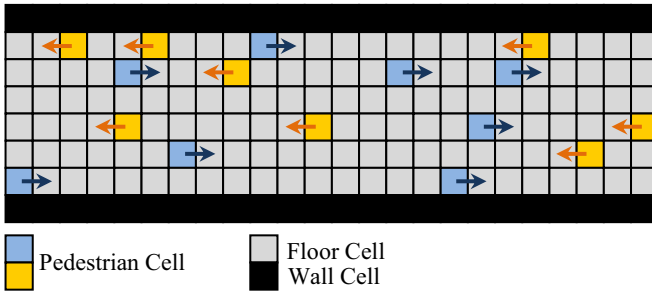


Fig. 1. Simulation Space and the Meaning of Each Cells

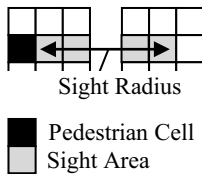


Fig. 2. Sight Area

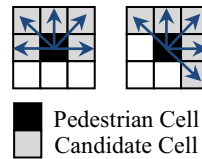


Fig. 3. Position of Candidate Cell

2.2 Learning Algorithm

Individual learning procedure had 3 steps; sight area recognition, checking situation of local area after moving, and updating situation pattern.

Step 1: Sight Area Recognition

The situation of cells in the sight area should be evaluated by the Pattern number expressed by 8 digit-number described in previous section.

Step 2: Checking Situation of Local Area after movement

After all pedestrians had moved, the Experienced value should be evaluated for each candidate cell, that is, the number of vacant continuous cells (α) and the sum of pedestrians in the candidate cells (β). The number of the pedestrians in the candidate cell did not include the marked pedestrian.

Step 3: Updating Situation Pattern

After finding the situation pattern which had the same pattern number from step 1, add the situation in the step 2 to each experienced value in the situation pattern. The each experienced values were updated in the situation pattern.

After the individual learning procedure, the information was shared by the all pedestrians walking into the simulation space at the entrance cell. The shared information was the pattern 8 digit-number and the average of the Experienced values (α , β and γ) for five candidate cells. The shared information might be updated when a pedestrian went out from the exit, and the new information would be shared by other pedestrians.

2.3 Local Neighbor and Transition Rules

2.3.1 Selection of the Candidate Cell

The cell where a pedestrian might move in the next time step was selected from 5 candidate cells. Evaluation value was setup to each candidate cell, and the cell which had the largest evaluation value was selected to move. Evaluation values of 3 candidate cells (front cell, left front, and right front) were compared at first. If more than 2 cells had equal evaluation value and front cell was included, the front one might be selected. If front cell was not included, either left or right front one was selected at random. If the 3 evaluation values were all zero, either of left cell or right cell was selected to move.

2.3.2 Update Information of Pedestrians

In each time step, heading degree, current degree were updated. Heading degree was updated after sight area recognition. The guide sign should be recognized at first in the sight area. If the distant between the guide sign and the goal was less than a certain value, the heading degree was updated to the degree to the guide sign.

At the same time, the current degree was updated after sight area recognition. The direction was determined by the vector of avoiding other pedestrian $\vec{a}_{pedestrian}$ and the vector of referring guide signs \vec{a}_{guide} . The vector of avoiding other pedestrian was determined in reference to the direction of a nearest pedestrian in opposite direction. This vector was define $\vec{a}_{pedestrian}$ as a unit vector which has degree as 180 degree added to the current degree of the opposite directional pedestrian. The vector of referring guide signs was defined from all guide signs which had the same goal to the pedestrian in the sight area.

3 Simulation Results and Discussions

In this chapter, initial and boundary conditions are briefly introduced. Two examples of pedestrian flow simulation, the counter flow along the straight pathway and on the corner were shown.

The counter flow was simulated in a straight pathway with 2 entrances. The pedestrians entered from each side of entrance and walked out from the other side of exit. In this simulation, the number of crash of pedestrians on the pathway was counted to check the effectiveness of learning procedure. The flow pattern of pedestrian was discussed in relation to the pedestrian density.

The corner flow in L-shaped concourse with 2 entrances was also simulated. In this simulation, some image files are shown to check the formation of rows by the proposed learning algorithm.

3.1 Basic Setting and Conditions

The basic setting, initial and boundary conditions of the simulation are shown as follows;

- A cell was a square with 75cm each side.
- A pedestrian could move 1 cell in each step.

- Entrance cell was specified in advance.
- Pedestrians walked into the simulation space from entrance cell at constant rate.
- If the number of pedestrians on the entrance cell was bigger than 0, new pedestrian could not walk into the cell.
- The width of the entrance is 10 cells. (It means that the width of the concourse is 7.5m)
- The number of crash was counted as the number of cells where the number of pedestrian was more than 1.

3.2 Counter Flow Simulation

The basic setting in the counter flow simulation is as follows;

- The pathway had length of 100 cells and width of 10 cells.
- The entrance and the exit were placed at the end of pathway in each case of simulation.

3.2.1 Effect of Learning Procedure

The number of times of crash for all pedestrians in the simulation space along time procedure is shown in Fig. 4, comparing the case with learning function and that without learning. The pedestrian walked into the pathway by the probability of 6% at the entrance cells. In Fig. 4, the number of times of crash was averaged by 1,000 times, because the crash number scattered widely in each simulation.

There seemed no crash up to 50 steps, because the groups of pedestrians from both entrance did not meet to each other. After the step 50, the number of crash quickly increased, and around the step 100 to 200, the number of crash came to the peak. This is because there were pedestrians coming into the pathway at the probability of 6%, and the rate of pedestrians walking into the simulation space decreased.

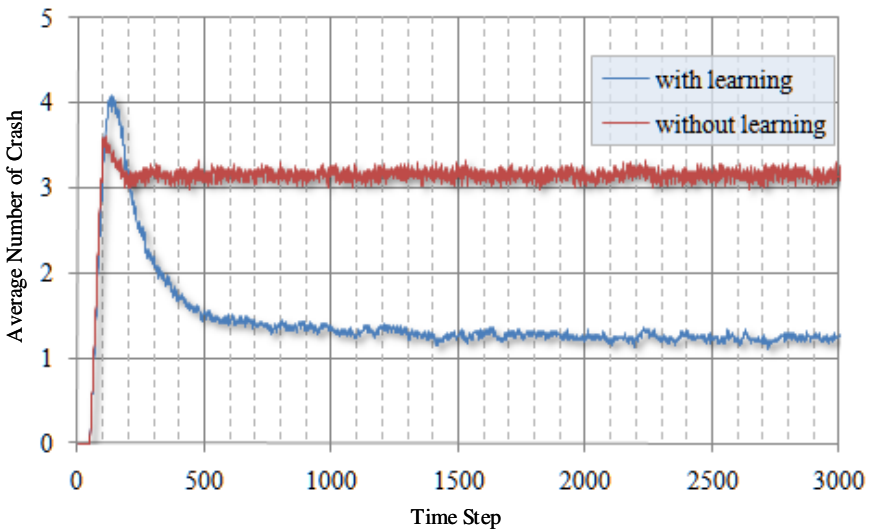


Fig. 4. The number of crash against time step by percentage of 6%

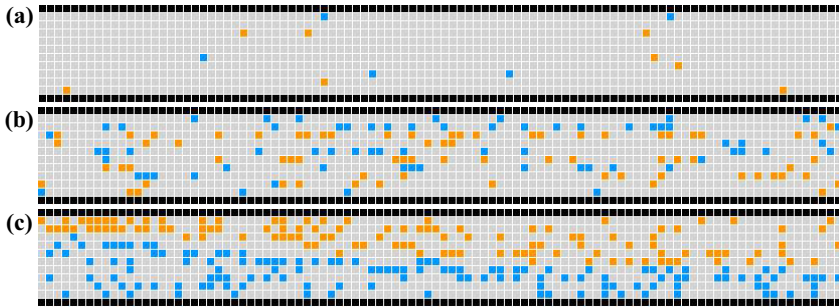


Fig. 5. The pedestrian flow pattern (a) 0.6%, (b) 7.0%, (c) 14.0%

The peak of learned result came after the result without learning and the peak value was larger than that without learning. This may be caused by the movement of pedestrians to avoid crashing under the condition of insufficient learning. The crash number quickly decreased after about the step of 140 and took steady value of a little over 1.0. On the other hand, the result without learning converged to around 3.0, which showed the effectiveness of learning procedure proposed in this paper.

3.2.2 Pattern of Pedestrian Flow on Straight Pathway

The flow patterns on the straight pathway are shown in Fig. 5, at different rate of pedestrian walking into the simulation space of 0.6, 7.0 and 14.0%. The simulation was performed with learning procedure.

At low density of pedestrian, people walked through the pathway on their own way, and there appeared no distinct pattern. On the contrary, as the density was increased, the pedestrians on the pathway tended to follow other persons in order to avoid crashing, which followed formation of the row of pedestrians.

The row formation is considered to be characterized by the number of rows and the number of people in each row. It is not easy to count the number of column in each scene of animation, because the length and width of a column varies in each time step on the simulation. Then, in reference to Fig. 6, the number of rows and the average number of people in the rows were estimated.

At first, a specific point is calculated on each row of cells. Each person walking from left to right corresponds to +1, and the person from right to left to -1. By adding the each point in every row of cells, the total point may be calculated for each row of cells. There may appear some boundaries of total points where the sign of point changes from positive to negative or vice versa. The number of rows may be characterized by this number of boundaries in the pathway⁷. There are two rows in the example scene shown in Fig. 6.

In this way, the number of rows were estimated in the results shown in Fig. 5. The relationship between the number of rows in the pathway and the density of pedestrian is shown in Fig. 7. The number of rows were estimated by the average of 10,000 steps. As the density was increased, the number of rows increased and showed a peak around 0.15-0.2 person/m². After the peak, the number of rows decreased and

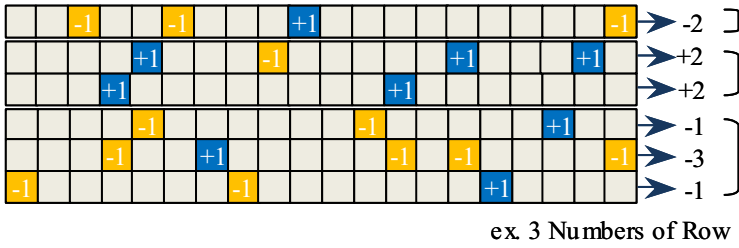


Fig. 6. Estimation of Number of Row

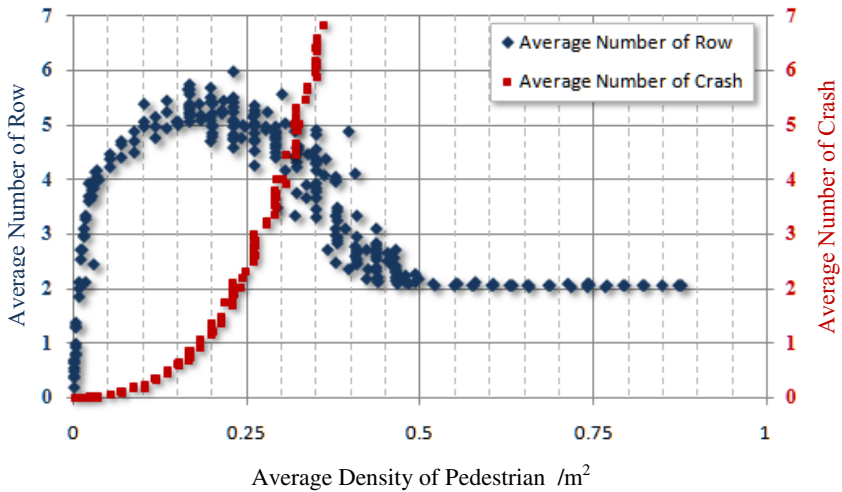


Fig. 7. Number of Rows against the Pedestrian Density

converged to around 2.0 at the density of 0.5 person/m², which means the counter flow was divided into upstream and downstream clearly.

At the same time, the average number of crash is shown in Fig.7. The peak of the number of row located at the density of 0.15 to 2.0 persons/m² might have some relation to the number of crash. It should be noted that, pedestrians on a passage had tendency to make crash in tapering rows, and to make rows wider for avoiding crash. It resulted in two-layered flow in opposite direction as shown in Figs.5(b) and (c).

3.2.3 Flow Pattern on a Corner

The flow pattern of pedestrians on a corner is shown in Fig. 8. In the result without learning, there appeared no distinct pattern, and people walked along the pathway crashing to each other. The simulation results with learning showed dynamic pattern formation of pedestrians. A pedestrian followed another pedestrian which induced a crowd flow, and that, the flow swayed dynamically in the width of pathway. The streams changed from inside to outside and vice versa.

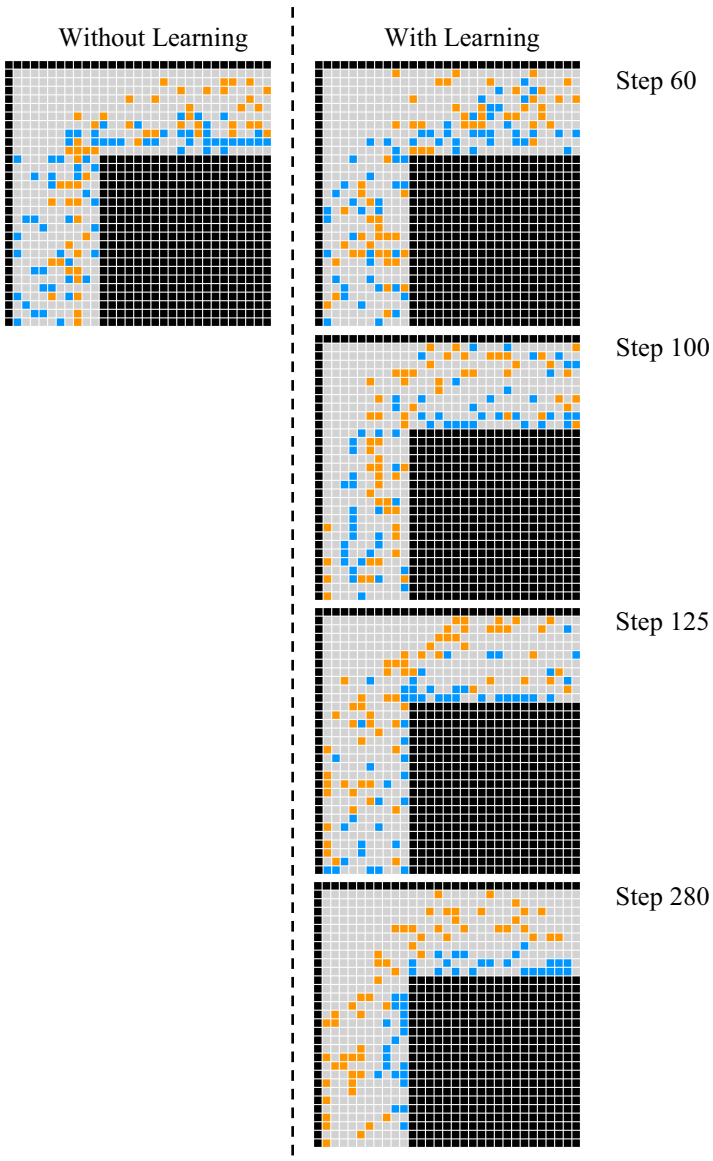


Fig. 8. Flow Patterns on a Corner

4 Conclusions

In this paper, CA was applied to the simulation of pedestrian flow, and a learning procedure to local neighbor and transition rules was proposed. The learning procedure was based on the number of continuous vacant cells in front of the target cell to proceed, and the number of pedestrian on the target cell. With this simple procedure,

pedestrians show the tendency to avoid crashing and to make rows on the pathway. Though there may exist various proposals of learning technique to model phenomena relating to human activities, one of the best and simplest ways to model them may be tracing the way how we think and behave.

References

1. Wolfram, S.: Cellular Automata. Los Alamos Science, 2–21 (Fall 1983)
2. Gutowitz, H.: Cellular Automata - Theory and Experiment, pp. 254–270. MIT Press, Cambridge (1990)
3. Mehta, A.: Granular Matter - An Interdisciplinary Approach, pp. 85–120. Springer, Heidelberg (1994)
4. Helbing, D., Farkas, I., Vicsek, T.: Simulating Dynamical Features of Escape Panic. *Nature* 407, 487–490 (2000)
5. Shiraishi, T., Morishita, S., Gavin, H.P.: Estimation of Equivalent Permeability in MR Fluid Considering Cluster Formation of Particles. *Transactions of ASME, Journal of Applied Mechanics* 71(2), 201–207 (2004)
6. Morishita, S., Nakatsuka, N.: Simulation of Emergency Evacuation by Cellular Automata. In: *Proceedings of 6th International Conference on Complex Systems*, pp. 92–97 (2002)
7. Narimatsu, K., Shiraishi, T., Morishita, S.: Acquisition of Local Neighbor Rules in the Simulation of Pedestrian Flow by Cellular Automata. In: Sloot, P.M.A., Chopard, B., Hoekstra, A.G. (eds.) *ACRI 2004. LNCS*, vol. 3305, pp. 211–219. Springer, Heidelberg (2004)

On Influencing of a Space Geometry on Dynamics of Some CA Pedestrian Movement Model

Ekaterina Kirik^{1,2}, Tat'yana Yurgel'yan², and Dmitriy Krouglov^{1,3}

¹ Institute of Computational Modelling SB RAS,
Krasnoyarsk, Akademgorodok, Russia, 660036
kirik@icm.krasn.ru

² Siberian Federal University, Krasnoyarsk, Russia

³ V.N. Sukachev Institute of Forest SB RAS, Krasnoyarsk, Russia

Abstract. In this paper we show an effect that a shape of way contributes to dynamics of one Cellular Automata pedestrian movement model. The fundamental diagrams for a closed and strait pathes are presented and discussed.

Keywords: pedestrian dynamics; transition probabilities; fundamental diagram.

1 Introduction

Here we present some investigation of dynamics of our model. The model is stochastic discrete CA model and supposes short-term decisions made by the pedestrians [1]. A possibility to move according the shortest path and the shortest time strategies are implemented to the model. From the comprehensive theory of pedestrian dynamics [2] such model may be refereed to tactical level.

It is obvious that a shape of a way influences on dynamics of people flow in real life. Here we focus on the influence of turns. The fact is that the pedestrian flow velocity goes down on turns; and model should be able to reproduce it. We investigated the realization of the same effect in our pedestrian movement model. The people flows were simulated under approximately constant densities on a straight path and a closed path. Differences between two cases were investigated comparing fundamental diagram.

In the next section the model is presented. Section 3 contains description of the case study and results obtained.

2 Description of the Model

2.1 Space and Initial Conditions

The space (plane) is known and sampled into cells $40cm \times 40cm$ which can either be empty or occupied by one pedestrian (particle) only (index $f_{ij} = \{0, 1\}$). Cells may be occupied by walls (index $w_{ij} = \{0, 1\}$) and other nonmovable obstacles.

The model imports idea of a map (static floor field S) from floor field (FF) CA model [3] that provides pedestrians with information about ways to exits. Our field S increases radially from exit cells. It doesn't evolve with time and isn't changed by the presence of the particles.

A target point for each pedestrian is the nearest exit. Each particle can move to one of four its next-neighbor cells or to stay in present cell (the von Neumann neighborhood) at each discrete time step $t \rightarrow t + 1$; i.e., $v_{max} = 1[step]$.

A direction of the movement of each particle at each time step is random and determined in accordance with the distribution of transition probabilities and transition rules.

2.2 Update Rules and Transition Probability

A scheme typical of the stochastic CA models is used. At the first stage, some preliminary calculations are made. Then, at each time step the transition probabilities are calculated, and the directions are selected. In the case, when there are more than one candidate to occupy a cell, a conflict resolution procedure is applied. Finally, a simultaneous transition of all the particles is made.

In our case, the *preliminary step* includes the calculation of FF S . Each cell $S_{i,j}$ stores the information on the shortest discrete distance to the nearest exit.

The probabilities of movement from cell (i, j) to, e.g., up neighbor is¹

$$p_{i-1,j} = N_{i,j}^{-1} \exp[k_S \Delta S_{i-1,j} - k_P F_{i-1,j}(r_{i-1,j}^*) - k_W (1 - \frac{r_{i-1,j}^*}{r}) \tilde{I}(\Delta S_{i-1,j} - \max \Delta S_{i,j})] (1 - w_{i-1,j}); \quad (1)$$

where

- $N_{i,j} = \tilde{p}_{i-1,j} + \tilde{p}_{i,j+1} + \tilde{p}_{i+1,j} + \tilde{p}_{i,j-1}$;
- $\Delta S_{i-1,j} = S_{i,j} - S_{i-1,j}$, $k_S \geq 0$ is the (model) field S sensitive parameter (the higher k_S , the better directed the movement);
- $r > 0$ is the visibility radius (model parameter) representing the maximum distance (number of cells) at which the people density and obstacles influence on the probability in the given direction;
- $r_{i-1,j}^*$ is the distance to the nearest obstacle in the given direction ($r_{i-1,j}^* \leq r$); the people density lies within $0 \leq F_{i-1,j}(r_{i-1,j}^*) \leq 1$;
- k_P is the (model) people sensitivity parameter which determines the effect of the people density, the higher parameter k_P , the more pronounced the shortest time strategy;
- $k_W \geq k_S$ is the (model) wall sensitivity parameter which determines the effect of walls and obstacles.

¹ Probabilities $p_{i,j+1}, p_{i+1,j}, p_{i,j-1}$ are calculated similarly. $p_{i,j} = 0$: the probability of retaining the current position is not calculated directly. Nevertheless, the decision rules are organized so that such opportunity could be taken.

The decisions rules are the following:

1. If $N_{i,j} = 0$, motion is forbidden.
2. If $N_{i,j} \neq 0$, target cell $(l, m)^*$, $(l, m)^* \in I = \{(i - 1, j), (i, j + 1), (i + 1, j), (i, j - 1), (i, j)\}$ is chosen randomly using the transition probabilities.
3. (a) If $N_{i,j} \neq 0$ and $(1 - f_{l,m}^*) = 1$, then target cell $(l, m)^*$ is fixed.
 (b) If $N_{i,j} \neq 0$ and $(1 - f_{l,m}^*) = 0$, then the cell $(l, m)^*$ is not available as it is occupied by a particle. In such case $p_{i,j} = \sum_{(y,z) \in I: (1-f_{y,z})=0} p_{y,z}$ and $p_{y,z} = 0 \forall (y, z) \in I : (1 - f_{y,z}) = 0$. Again, the target cell is chosen randomly using the transformed probability distribution.
4. Whenever two or more pedestrians have the same target cell, movement of all the involved pedestrians is denied with probability μ . One of the candidates moves to the desired cell with the probability $1 - \mu$. The pedestrian allowed to move is chosen randomly.
5. The pedestrians that are allowed to move perform motion to the target cell.
6. The pedestrians that appear in the exit cells leave the room.

The above rules are applied to all the particles at the same time; i.e., parallel update is used.

3 Case Study

To investigate the contribution of turnes to model dynamics we use two case studies, see fig. 1. The first path is strait; the other one is closed path. A set of densities was considered. During each experiment the initial density was kept approximately constant.

In all experiments we investigate the directed movement $k_S = 4$; the attitude to walls is "loyal" ($k_W = k_S$).

The simplest type of the way, the strait path, supposes that strategy of the shortest path coincides with the shortest time strategy for the whole way. Geometry of the way does not influence on the movement, and the shape of the

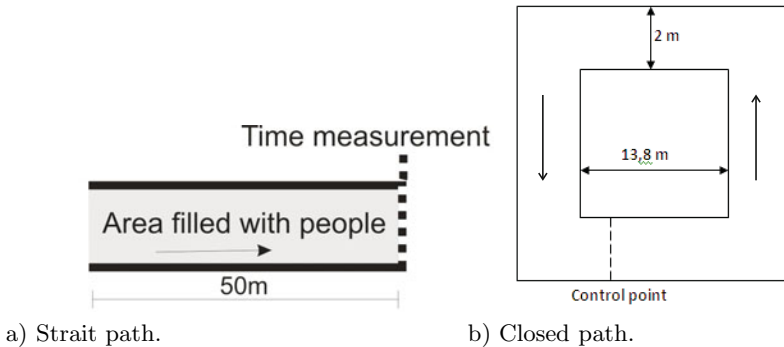



Fig. 1.

flow and velocity are only determined by the density. To realize only the shortest path strategy the model density sensitive parameter k_P has to be low ($k_P < k_S$).

If there are turns on the way, congestions appear before turns (depending on density), and some people start to use detours facilities (that means to follow the shortest time strategy) and not to wait when the shortest path will free. As a result the average velocity and flow go down. In the model the shortest time strategy is pronounced under $k_P > k_S$. The mechanism is the following. If the shortest path direction has a high density, $F(r^*) \approx 1$, the probability of this direction goes down. At the same time, the probability of direction(s) that are more favorable to movement ($F(r^*) \ll 1$) rises, and the detours around high-density regions are made. One can say that the model is density adjustable.

In figures 2a, 2b the fundamental diagrams presented for strait and for closed pathes correspondingly. Comparing figures one can see that the flow  goes down (approximately in half) from the strait path case to the closed one. Shapes of the fundamental diagrams change. Maximum of the flow shifts to lower densities. Thus, general expectations are realized.

We tested different sets of parameters. These sets reproduce the different people movement: from using only one strategy (the shortest path) when $k_P < k_S$ to combining both strategies if $k_P > k_S$.

In fig. 2a one can see that the flows are the highest and approximately coincide for 3 sets of parameters ($k_S = 4, k_P = 2, r = 1$; $k_S = 4, k_P = 4, r = 1$; $k_S = 4, k_P = 4, r = 10$). In all of this cases the shortest path strategy is mainly reproduced by the model; the influence of the people density sensitive term is reduced to minimum by low parameter k_P .

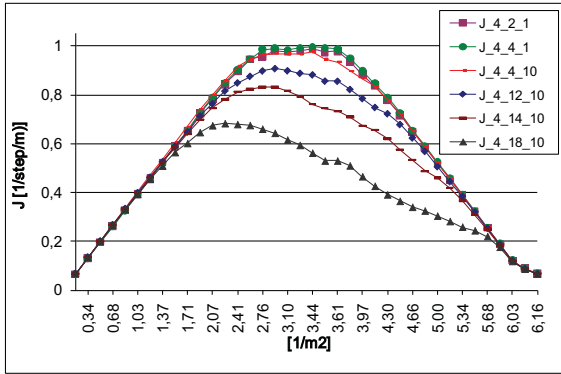
The other curves in fig. 2a give flows for cases when the the shortest time strategy is already reproduced by the model. But the type of the path does not suppose using of this strategy. And realizing of the shortest time strategy delivers some disturbance to the directed movement, average velocity of the flow slows down, and this results in the lower flow.

Note that for wide range of low densities and for high densities all sets of model parameters give the same flow. This says that for such type of way the model is sensitive to model parameters only under middle flow density.

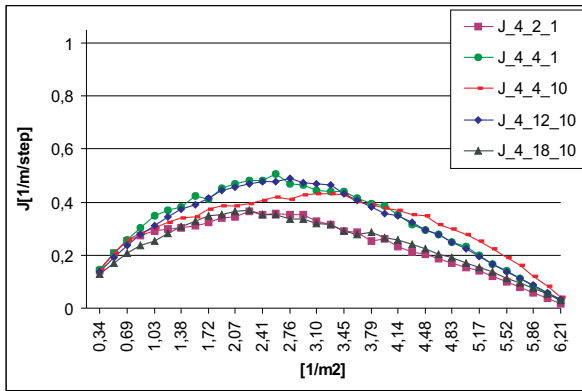
At the same time a comparison of figures 2a and 2b shows that for the closed path starting with the lowest densities the model is sensitive to the parameters. Curves in figure 2b diverge for the whole range of densities and approximately coincide only for extreme density values. But value of maximal divergence is considerably less then in fig. 2a.

Moreover dynamics of the model, on the whole, is very sensitive to the shape of way. Only for $\rho < 0,75[1/m^2]$ flows for parameters considered in figure 2a and figure 2b approximately coincide. Starting with $\rho > 0,75[1/m^2]$ presence of turns results in a slowing down of the velocities and flows (approximately in half).

² We use specific flow $J = 1000/T_{st}/2 [1/step/m]$, where T_{st} – number of steps that 1000 particles need to cross the control line under given density.



a) Strait path.



b) Closed path.

Fig. 2. Fundamental diagram for different sets of parameters k_S, k_P, r ($J_{k_S k_P r}$)

Interesting facts are: the most divergent curves from fig. 2a ($J_{4_2_1}$ and $J_{4_18_10}$) approximately coincide in the closed path case; the most coincident curves from fig. 2a ($J_{4_2_1}$ and $J_{4_4_1}$) are the most divergent in fig. 2b.

The first fact may be explained in the following way. Parameters $k_S = 4, k_P = 2, r = 1$ and $k_S = 4, k_P = 18, r = 10$ deliver opposite extreme strategies of movement (see above); and for the straight path this gives expected very divergent curves. For the closed path these opposite properties gives coincident the lowest flows because the type of the way implies the combining of the strategies.

4 Conclusion

At the moment we have no appropriate real data for the similar closed path to compare with. But simulation results obtained show the expected decreasing of the flow comparing the straight and the closed paths. We believe that specific

feature of CA model, i.e., discreteness of the space and the von Neumann neighborhood, gives some contribution to the decreasing. But nevertheless the proper model “senses” the shape of the way.

At the same time simulation results show that model parameters play important role. Of course the fundamental diagram could not depict the all variety of the difference in model dynamics for different parameters and type of ways, and more criterions should be investigated for thorough identifying the all features of the model dynamics. But time and spatial adaptation of model parameters becomes clear to make the model geometry adjustable.

Acknowledgment

This work is supported by the Russian Government programme on Fire Safety in Russian Federation, contract 09.0708.11.014.

References

1. Kirik, E., Yurgel'yan, T., Krouglov, D.: The Shortest Time and/or the Shortest Path Strategies in a CA FF Pedestrian Dynamics Model. *Journal of Siberian Federal University, Mathematics and Physics* 2(3), 271–278 (2009)
2. Schadschneider, A., Klingsch, W., Kluepfel, H., Kretz, T., Rogsch, C., Seyfried, A.: Evacuation Dynamics: Empirical Results, Modeling and Applications. In: *Encyclopedia of Complexity and System Science*. Springer, Heidelberg (2009)
3. Schadschneider, A., Seyfried, A.: Validation of CA models of pedestrian dynamics with fundamental diagrams. *Cybernetics and Systems* 40(5), 367–389 (2009)

The Dynamic Distance Potential Field in a Situation with Asymmetric Bottleneck Capacities

Tobias Kretz

PTV AG, Stumpfstraße 1, D-76131 Karlsruhe
Tobias.Kretz@PTV.De

Abstract. This contribution discusses the application of a fast and sloppy solution of the Eikonal equation – namely the dynamic distance potential field – for the simulation of the flow of a group of pedestrian agents through two bottlenecks with different capacity (width) but identical walking distance toward the destination. It is found that using the method leads to a better distribution of agents on the two corridors.

1 Introduction

It is a wide spread paradigm in the modeling of pedestrian dynamics [1] to include three basic elements in the model: 1) a pedestrian wants to move (spatially) closer to the destination; 2) a pedestrian evades other pedestrians; 3) a pedestrian evades obstacles. There's the – often unsaid – implicit assumption that in combination these elements act together to make the simulated pedestrians (*agents*) move not on the shortest but the quickest path to an extent that corresponds to real pedestrians' behavior. This, however, in many cases does not work as intended as a situation as simple as the flow around a corner shows [2,3].

1.1 Quickest Path for Road Networks

For vehicular traffic it is a central problem to find the user equilibrium of a given transportation network and demand [4,5]. For a system in user equilibrium it is impossible for a single driver to reduce the travel time by changing the route. For a single driver this is equivalent to finding the quickest route under consideration of the restrictions imposed by the presence of all other drivers.

Finding the equilibrium is simpler for a road network than for a pedestrian “areawork” as the route choices are discrete and the number of routes is finite – although the generation of the contemplable routes is not trivial [6]. In a sense for vehicles the routes can be used as input while they (the trajectories) are a result of the simulation of pedestrians. For vehicular dynamics the equilibrium is found by iterative simulation or calculation. Apart from methods that lead to results close to the equilibrium in some situations there's no way to get rid of the iterative approach. The iterative approach is possible as the routes stay exactly the same from one iteration step to the other. On the contrary pedestrian

trajectories are individual properties that usually do not re-occur identically in subsequent iteration steps.

For this reason the method discussed in this contribution in fact addresses the problem of an individual to try to minimize remaining travel time but it refrains from iterative simulation. Therefore it is obvious right from the beginning that the result can not be an equilibrium in the rigid sense. The aim is that by using the method the system obviously – already visually – is closer to an equilibrium state than when the method is not used.

1.2 Quickest Path in Continuous Space

Fermat observed that light travels on the path of least time, provided that the starting and end points A and B are fixed. In environments with a variable speed of light (different materials, but also a spatially continuously changing concentration of a solvent in a solution) light therefore can considerably deviate from the shortest path – the most famous consequence being Snell's law.

Put in a more general mathematical formulation light follows the gradient of the solution of the Eikonal equation [7,8]

$$|\nabla T(\mathbf{x})|^2 = \frac{1}{|v(\mathbf{x})|^2} \quad (1)$$

with $v(\mathbf{x})$ as travel speed at location \mathbf{x} and $T(\mathbf{x})$ as remaining travel time toward the destination B .

The Fast Marching Method is a well established and widely used numerical method to solve and integrate the Eikonal equation [9,10]. While it is considered to be fast, it still has a major impact on computation times, if it is integrated into a model of pedestrian dynamics for time-step-wise recalculation. Nevertheless it has already been used in a model of pedestrian dynamics [11]. There the effects of distance, travel time and discomfort are combined to calculate one single potential field which then exclusively determines the motion of the agents. Another application of the Fast Marching Method for the simulation of pedestrians has recently been presented in [12]. Most applications there are static such that computation time is a less important issue, but it is shown that the method can easily be transferred to other lattice geometries, as in this case the model of pedestrian dynamics is formulated on a hexagonal lattice.

The method used in this work is simpler and accepts limited precision in the solution of the Eikonal equation in exchange for quicker computation.

Solving the Eikonal equation numerically means calculating values for nodes of a mesh. This mesh can be chosen according to the spatial structure of a CA model of pedestrian dynamics. While using numerical solutions of the Eikonal equation is not limited to CA models of pedestrian dynamics it appears therefore to be especially suited for them.

No matter what method is used for solving the Eikonal equation and no matter how small its errors are, one problem remains: the solution is calculated instantaneously. This means that the value of the field $T(\mathbf{x})$ at position \mathbf{x}_0 of an agent A depends on the conditions downstream toward the destination as

they are *now*. Relying on a solution of the Eikonal equation to calculate quickest paths means that either the conditions (the travel speeds) are constant – which typically is the case for light – or that one knows about how the conditions will develop. For pedestrians the conditions typically will develop, as the distribution of pedestrians will develop and as the local distribution of pedestrians is crucial for the travel speed one can assume to exist at a certain location. This is again different for light: the distribution of light (photons) influences the propagation of light only marginal.

To find the quickest path therefore agent *A* would have to be informed on the pedestrian density as it will be when he is there *at some later time*. Yet to know about this one would be in need to know the results of the simulation. The only way to escape this dilemma would be a converging iterative process, which is not possible in a straight-forward way for reasons stated above.

As real pedestrians do not have perfect information on the conditions on their remaining path, they normally are not able to choose the quickest route as well. Nevertheless one must be aware that using a solution of the Eikonal equation as one of the determinants of motion will not necessarily in general yield perfectly realistic walking directions. Again here the reasoning of the last paragraph of the preceding applies: the intention is to have not a perfect but a better agent behavior by applying the method.

Recently the Dynamic Distance Potential Field was introduced as a heuristic method to make simulated pedestrians value the quickest path better compared to the shortest path [13]. It has so far mainly been tested in geometries, where an alternative longer path has the same capacity (bottleneck width) as the shorter path [14,15]. In the remainder of this work at first the method is sketched and then applied to an example geometry that is basically symmetric in distances, but asymmetric in capacities. This is to be understood as completion of previous works with examples with symmetric capacity but varying walking distances.

2 The Dynamic Distance Potential Field

For the calculation of the dynamic distance potential field an additional lattice is used. It shares spatial properties with the lattice of the cellular automata model of pedestrian dynamics (here the F.A.S.T. model [16,17,18] was used, but the method can be combined with any other CA model of pedestrian dynamics just as well). The values for this lattice are calculated by propagating a front from the destination outward. When propagating over lattice cells which are free (not occupied by a pedestrian) a value of 1 is added, when propagating over occupied cells a value $s_{add} > 1$ is added. This is done two times: once the propagation only floods over common edges and second over common corners as well. In the former case the resulting field is called D_M (with the *M* from *Manhattan metric*), in the latter case D_C (with the *C* from *Chebyshev metric*). The dynamic distance potential field D_{V_1} is a combination of both:

$$D_{V_1} = \sqrt{D_C^2 + (D_M - D_C)^2} \quad (2)$$

This combined field exhibits smaller errors than the two composite fields alone [19]. To reduce the error further the value influencing the motion is not the field itself but the difference between the field and the field calculated for the geometry without pedestrians

$$\Delta D_{V_1}(t) = D_{V_1}(t) - D_{V_1}(\text{empty}). \quad (3)$$

3 Geometry

The geometry is an alternation of RIMEA's test case 11 (RTC 11). While in RTC 11 1,000 agents have to leave a room via one of two equally sized doors, of which one is more remote than the other, here 1,000 agents have to leave a room by one of two doors that are equally far away compared to the starting area of the pedestrians, but of which one is twice as wide as the other. Of this basic structure two variants have been considered: one with short (40 cm) and one with long (10 m) bottlenecks. See figure 1.

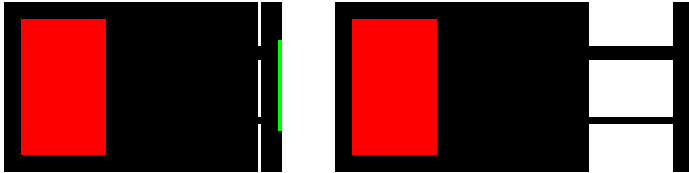


Fig. 1. Investigated geometries: 1,000 agents start on the red area, which has 6.25 x 10 m. The bottlenecks are 80 resp 160 cm wide and 0.4 resp 10 m long. The exit is marked with the green light at the right side.

4 Simulation Model and Basic Considerations

The study was done using the F.A.S.T. model for simulation. In the F.A.S.T. model agents move on a discrete regular lattice with a distance of 40 cm between the nodes. In other words there are square shaped cells with an edge length of 40 cm and each cell can be occupied by at most one agent. An agent is always located exactly on one cell.

This implies that the starting area is densely packed with 25 x 40 agents. As the central axes of both bottlenecks are placed symmetric to the starting area, the horizontal symmetry axis of the starting area is a bit closer to the inner edge of the wider bottleneck than to the narrow bottleneck. One row of cells on the starting area is equally far away from the inner edges of both bottlenecks. From this one would expect a 487.5 : 512.5 distribution of agents on the two bottlenecks, if it were only for the shortest path.

If one would only have the agents of one bottleneck group as a waiting crowd half-circle shaped around the bottleneck, one sees that a part of the half-circle is closer to the other bottleneck. Those agents then would use this other bottleneck. Thus, energy (in the static potential) minimizing effects can lead to a

re-distribution with regard to the bottlenecks. This does not happen to full extend, as both groups are there at the same time. But as the flow through the wider bottleneck is larger, the effect exists to some extend. If the effect would be present in full extend in the direction narrow to wide bottleneck, by simple geometric considerations one would expect a distribution of about 393 : 607.

The distribution according to capacities trivially is 333.3 : 666.7.

5 Results

The number of agents that use the narrow corridor (see figure 2) is generally larger for the short bottleneck. This is a consequence of the effect described in the previous section. The whole process needs less time at the short bottleneck (as can be seen in figure 3). Therefore the re-distribution is less pronounced at the short bottleneck compared to the long. With the dynamic distance potential field switched off ($s_{add} = 1$ or $k_{Sdyn} = 0$), the number of agent walking the long narrow corridor is even smaller than calculated in the previous section and significantly larger than this value in the case of short bottlenecks.

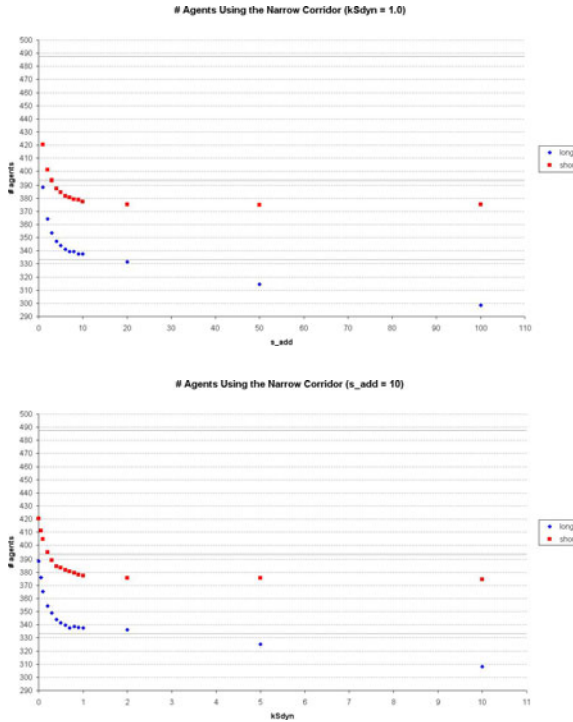


Fig. 2. Asymmetry plot, resp. number of agents using the narrower corridor. The additional horizontal lines mark the special values discussed in section 4. Each data point is the average of 100 simulation runs.

With increasing s_{add} and increasing k_{Sdyn} the number of agents dwindles that use the narrow bottleneck and this as much for the short as the long bottleneck. Only for very large values of k_{Sdyn} and s_{add} there is a difference: At the short bottleneck there appears to be a saturation, while at the long bottleneck the number reduces more and more, even below the value of 333.3. This is an effect of the number of barriers that can build in the long bottleneck. Whenever there is a “bridge” of occupied cells from one edge of the bottleneck to the other, the dynamic distance potential field cannot flow around the agents but must pass them. Then s_{add} determines the value of up-potential cells and not the size of the block of occupied cells. For the long bottleneck theoretically there are 25 occasions for this, for the short bottleneck, there is only one.

The total evacuation time (figure 3) shows a minimum in the case of a long bottleneck at moderate values of s_{add} and k_{Sdyn} and then clearly increases again. One reason is that less agents than in the capacity-balanced case pass the narrow bottleneck, but the increase begins at values of s_{add} and k_{Sdyn} that are even smaller than those for which the number of agents using the narrow bottleneck crosses this value downwards. For the short bottleneck the increase for large s_{add} and k_{Sdyn} is much smaller. The average individual egress times show the same tendency as the corresponding total evacuation times, but much less pronounced.

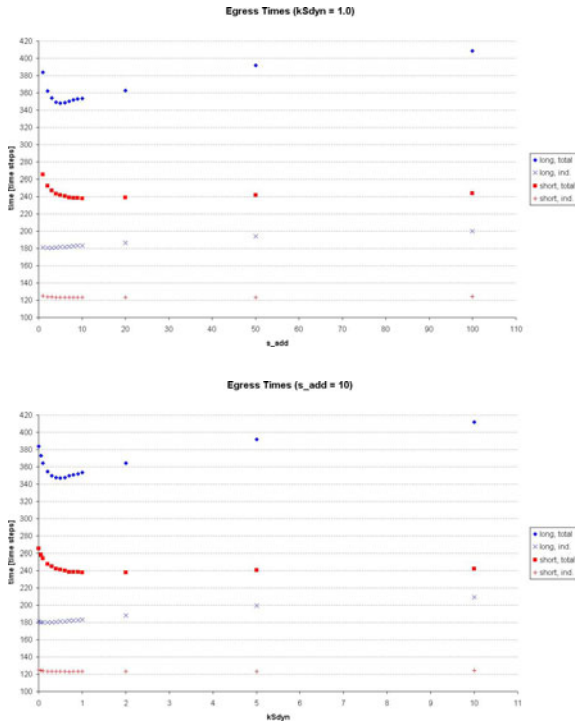


Fig. 3. Egress times. “total” is the evacuation time (time until the last agent has left) and “ind.” is the average of individual times to reach the exit.

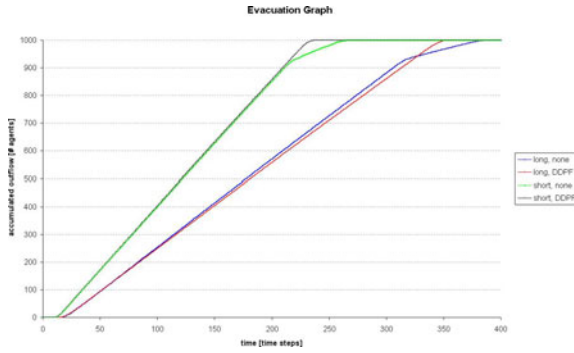


Fig. 4. The evacuation graph

All this can be understood looking at the evacuation graph (figure 4). With dynamic distance potential field the outflow rate (the slope of the graph) in the case of the long bottleneck during the long “steady state” period is not larger but smaller than without dynamic distance potential field. Only at the end the more balanced distribution of agents on the two bottlenecks leads to an earlier completion of the process. If one takes a very close look, for the case of the short bottleneck one can see that in contrast the outflow rate is larger with dynamic distance potential field than without. The reason is that in the F.A.S.T. model the dynamic distance potential field slightly increases the average speed of an individual agent in very low densities, but reduces the flow in densities at and right above capacity.

This is similarly illustrated by the global directed flow shown in figure 5.

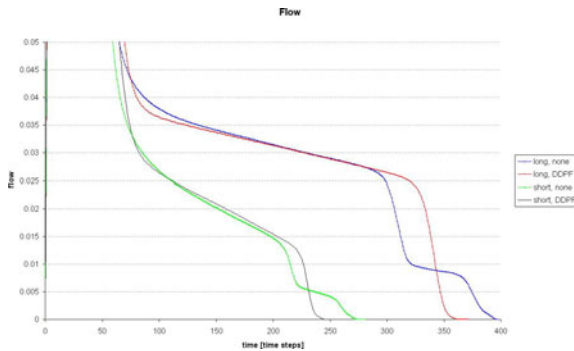


Fig. 5. Global directed flow: In this plot the flow is defined as sum over all agents of changes in the value of the static potential of their current compared to their position in the preceding time step. This is normalized by the size of the unobstructed area (the number of accessible cells). I.e. it is average destination directed speed times global density [20]. It can clearly be seen, how the better distribution on the two bottlenecks leads to a higher flow in the end, if the dynamic distance potential field is applied. For each of these functions 40,000 simulations have been done to get a smooth average.

6 Summary

The dynamic distance potential field method has been tested at a geometry with two almost equally long routes with different capacity. The method is able to distribute the agents more according to the bottlenecks' capacity. In the case of the short bottleneck the ratio of bottleneck usage comes closer to the ratio of widths and in the case of the long bottleneck it's even possible to overcompensate the asymmetry of widths. As this is an unwanted side-effect, it shows that one has to take care when setting the parameters k_{Sdyn} and s_{add} .

The effect on the evacuation time is comparatively small and the effect on individual egress times almost negligible. This is because even without influence of the dynamic distance potential field the distribution on the two bottlenecks is not as bad as it is, if one of two routes is longer (where it can easily be 0:100). The evacuation time becomes minimal at about $k_{Sdyn} \cdot s_{add} = 5$. For larger values it becomes larger again and can even increase above the case without dynamic distance potential field.

Future options which are currently in parts under development for the pedestrian simulation in VISSIM [21,22], include the following items:

Using the method in a spatially continuous model poses a number of technical challenges which do not exist when the lattice of agents' positions and the one of the dynamic distance potential field have identical properties.

Linked to this is the question, if it yields the best results, if only the directly occupied cells receive the value s_{add} or if the results can be improved if also for unoccupied locations as well density is calculated in one way or another [23,24] and for a non-zero density a value $s_{add} > 1$ is assigned to the cell. This would entail the need to set the dependence of speed on density and by that make the model more first order-like. As the dependence of speed on density is as well calculated by the model, the model could become inconsistent.

The parameter s_{add} can be calculated variably depending on if an agent is heading into the same or opposite direction as those who share the destination.

Recently a method [25] has been developed that reproduces the results of the Fast Marching Method and therefore has much smaller errors than the method used here. In principle the method is slower in calculation than the Fast Marching Method, but it is much more suited for parallel computation and exploitation of the specific abilities of GPUs.

References

1. Schadschneider, A., Klingsch, W., Klüpfel, H., Kretz, T., Rogsch, C., Seyfried, A.: Evacuation Dynamics: Empirical Results, Modeling and Applications. In: [26], p. 3142
2. Steffen, B., Seyfried, A.: Modeling of pedestrian movement around 90 and 180 degree bends. In: Proc. of Workshop on Fire Protection and Life Safety in Buildings and Transportation Systems (2009)
3. Rogsch, C., Klingsch, W.: Risk analysis with evacuation software how should we interpret calculated results. In: Interschutz (2010) (in press)

4. Wardrop, J.: Road Paper. Some Theoretical Aspects of Road Traffic Research. Proceedings of the Institute of Civil Engineers 1, 325–362 (1952)
5. Dafermos, S., Sparrow, F.: The traffic assignment problem for a general network. J. Res. Natl. Bur. Stand., Sect. B 73, 91–118 (1969)
6. Gentile, G.: Linear User Cost Equilibrium: a new algorithm for traffic assignment. Submitted to Transportation Research B (2009)
7. Bruns, H.: Das Eikonal. S. Hirzel (1895)
8. Frank, P.: Über die Eikonalgleichung in allgemein anisotropen Medien. Annalen der Physik 389 (1927)
9. Sethian, J.: Level Set Methods and Fast Marching Methods. Cambridge University Press, Cambridge (1999)
10. Kimmel, R., Sethian, J.: Computing geodesic paths on manifolds. In: PNAS, pp. 8431–8435 (1998)
11. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. In: Siggraph, p. 1168 (2006)
12. Hartmann, D.: Adaptive pedestrian dynamics based on geodesics. New Journal of Physics 12, 043032 (2010)
13. Kretz, T.: Pedestrian Traffic: on the Quickest Path. JSTAT P03012 (2009)
14. Kretz, T.: The use of dynamic distance potential fields for pedestrian flow around corners. In: ICEM, TU Delft (2009)
15. Kretz, T.: Applications of the Dynamic Distance Potential Field Method. In: Dai, S., et al. (eds.) TGF 2009, Shanghai. Springer, Heidelberg (2010)
16. Kretz, T., Schreckenberg, M.: F.A.S.T. – Floor field- and Agent-based Simulation Tool. In: Chung, E., Dumont, A. (eds.) Transport simulation: Beyond traditional approaches, pp. 125–135. EPFL press, Lausanne (2009)
17. Kretz, T., Schreckenberg, M.: The F.A.S.T.-Model. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 712–715. Springer, Heidelberg (2006)
18. Kretz, T.: Computation Speed of the F.A.S.T. Model. In: Dai, S., et al. (eds.) TGF 2009, Shanghai. Springer, Heidelberg (2010) (in press)
19. Kretz, T., Bönsch, C., Vortisch, P.: Comparison of Various Methods for the Calculation of the Distance Potential Field. In: Klingsch, W., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) PED 2008, Wuppertal, pp. 335–346. Springer, Heidelberg (2009)
20. Kretz, T.: Pedestrian Traffic – Simulation and Experiments. PhD thesis, Universität Duisburg-Essen (2007)
21. Helbing, D., Johansson, A.: Pedestrian, Crowd and Evacuation Dynamics. In: [26], p. 6476
22. PTV: VISSIM 5.30 User Manual, PTV Planung Transport Verkehr AG, Stumpfsstraße 1, D-76131 Karlsruhe (2010), <http://www.vissim.de/>
23. Helbing, D., Johansson, A., Al-Abideen, H.: Dynamics of crowd disasters: An empirical study. Physical review E 75, 46109 (2007)
24. Steffen, B., Seyfried, A.: Methods for measuring pedestrian density, flow, speed and direction with minimal scatter. Physica A (submitted)
25. Jeong, W., Whitaker, R.: A fast eikonal equation solver for parallel systems. In: SIAM Conference on Computational Science and Engineering (2007)
26. Meyers, R. (ed.): Encyclopedia of Complexity and Systems Science. Springer, Heidelberg (2009)

Solving the Direction Field for Discrete Agent Motion

Michael Schultz¹, Tobias Kretz², and Hartmut Fricke¹

¹ Institute of Logistics and Aviation, Technische Universität Dresden,
Hettner Str. 1-3, D-01062 Dresden, Germany
{schultz, fricke}@ifl.tu-dresden.de

² PTV Planung Transport Verkehr AG,
Stumpfstr. 1, D-76131 Karlsruhe, Germany
tobias.kretz@ptv.de

Abstract. Models for pedestrian dynamics are often based on microscopic approaches allowing for individual agent navigation. To reach a given destination, the agent has to consider environmental obstacles. We propose a direction field calculated on a regular grid with a Moore neighborhood, where obstacles are represented by occupied cells. Our developed algorithm exactly reproduces the shortest path with regard to the Euclidean metric.

Keywords: direction field, regular grid, Moore neighborhood, Euclidean metric, error compensation, flood fill, algorithm.

1 Distances and Directions

To give robots or simulated pedestrians (agents) their main direction of movement, for at least three decades there have been two main methods: one uses a set of navigation points to steer the agent around obstacles [1], the other one – which this contribution deals with – relies on a grid in which each grid cell holds the information on the walking distance to the destination and/or the direction to move on to be on the shortest or roughly quickest path considering the location of obstacles. This grid is often called a “potential” or a “distance look up table”. An early usage of the notion and method of a potential was made by Khatib [2] in robot motion planning. But this potential did not consider the location of obstacles but just held the bee-line distances from the location of the robot to the destination. This makes sense in the motion planning of *autonomous* robots, which only have a very limited knowledge of their environment and need an elaborate method to escape dead-ends anyway [3].

The most prominent and most widely used method that calculates Euclidean distances as a numerical solution of the Eikonal equation [4, 5] comparatively fast and with a comparatively small error is the Fast Marching Method (FMM) [6–8]. Concerning computation time the FMM shows optimal worst-case behavior and the relative error in general decreases with increasing distance from the destination, implying that with a finer grid the error can be reduced. With the change

of computing power progress from improving single CPU processing to multi-core computation, slight changes in the algorithm of the up-winding scheme have become useful and recently new methods as for example the Fast Sweeping [9] and the Fast Iterative Method [10] (FSM and FIM) have been introduced. For a study of computation times of such methods see [11] for example.

Apart from optimizing the numerical Eikonal equation solver algorithmically, it's also possible to trade exactness for computation speed and use simpler methods for the calculation. The two most prominent examples for this are a simple flood fill over common edges or common edges and common corners which lead to metrics in vector norms [1], $p = 1$ (Manhattan metric) or $p \rightarrow \infty$ (Chebychev metric) respectively.

$$d_p^n = \|x\|_p := \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \quad (1)$$

With Euclidean metric (i.e. vector norm $p = 2$) as correct solution, these methods lead to relative errors which remain constant over distance or even increase, which means that a finer grid size does not improve the precision arbitrarily. However, it is possible to reduce the error by making some slight modifications upon the simple flood fill methods [12]. Just for completeness we want to add that to our knowledge no other than ray tracing methods exist, that are able to reach in general cases the minimal error possible, which is given by the grid resolution [12]. However, with such methods compared to FMM, FSM or FIM normally one pays with a tremendous increase of computation time for a small gain in exactness.

In many models of pedestrian (or robot) dynamics only the desired walking direction (i.e. the gradient of walking distances) but not the walking distances themselves are inputs for the calculation of the dynamics. Therefore in this contribution we put forward a new method to calculate the (in terms of the Euclidean metric) exact walking directions without having to calculate the exact Euclidean walking distances. The method does neither rely on a computation time expensive sorting of distances of currently *active* cells, nor does it even has the need for the calculation of rather computation-time expensive functions (e.g. square roots). This is achieved on the expense that the method can only be used to calculate directions based on shortest distance and not shortest time [13-20] and that only the directions but not the distances are calculated exactly regarding to the Euclidean metric.

2 Algorithm

A common practice for creating a distance potential is a flood fill approach. The following *flood fill* algorithm describes the essential steps for the creation of the potential field (*breadth first search algorithm* applied for von Neumann neighborhood [21]).

- The target cell T is initialized with the distance $d_T = 0$, the other n cells get the distance $d_n = \infty$.

- Put T in a first-in, first-out queue Q .
- While cells available in Q do:
 - poll (get and remove) the first cell (C) from Q
 - set C as center cell
 - for all adjacent cells C_n :
 - calculate distance to the target cell regarding to the center cell $d_n = \min(d_n, d_C + \Delta d)$
 - if d_n changes, put C_n in Q

The distance between adjacent cells depends on the relative position (diagonal or horizontal/vertical). Using a geometry G , non-accessible cells (e.g. environmental obstacles) possess a distances of $\Delta d = \infty$. For Δd three different cases exist:

- ∞ , if C_n is a infrastructure cell,
- $\sqrt{2}$, if C_n is diagonal located regarding to C , and
- 1 , if C_n is horizontal/vertical located regarding to C .

In comparison to the Manhattan (quadrant I, see fig. 1), Euclidean (II) and Chebychev (III) metrics, the flood fill approach for Moore neighborhood points out a fourth distance metric (IV):

$$\|x\| := |\Delta x_i| + \sqrt{2} \min(|x_i|) . \tag{2}$$

As shown in fig. 1 the presented flood fill metric approximates the Euclidean metric in a roughly way and it tends to overestimate the distance (except at the grid symmetry axes). However, the flood fill approach is often used to create the potential field for agent navigation. The corresponding agent rule is: "Choose the cell with the closest distance to the target".

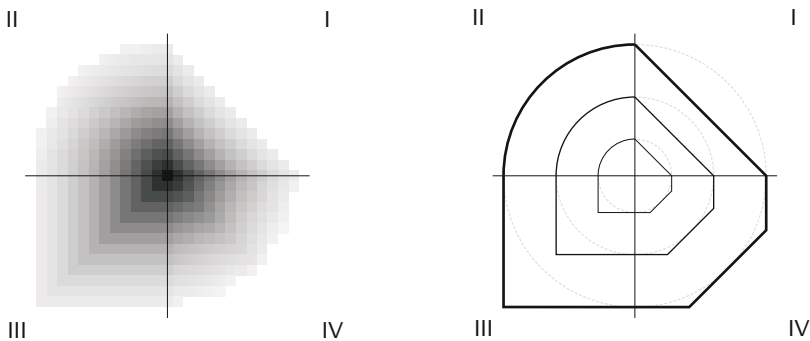


Fig. 1. Distance metric characteristics: transition from black to white corresponds with decreasing potential (left), equipotential lines (right)

3 Direction Field

Despite to the significant deficiency of the flood fill metric, we will demonstrate that this approach can be used for creating a precise direction field [22]. To create a simple direction field D (represents an array of motion vectors) the distance field is created for a test scenario (see fig. 2). Close inspections of the color-coded distance field at fig. 2 (right) already reveals the distance metric for the Moore neighborhood.

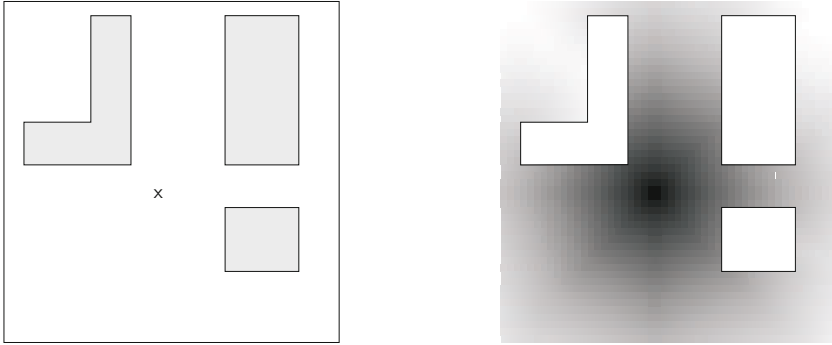


Fig. 2. Test scenario with a centered target and three obstacles (left) together with the corresponding color-coded distance field (decreasing target distance from white to black, right)

The flood fill algorithm stores the shortest target distance for each cell, whereas the cell specific distance always will base on a diagonal or horizontal/vertical located adjacent center cell. The location of the upstream center cell depends on the processing sequence of the adjacent cells, which is defined by the algorithm. To determine the direction field, the derived cell based motion vector points to the particular center cell. Detailed verifications show, that the characteristics of the simple direction field depends on the particular implementation (see fig. 3). First implementations determine the sequence of the adjacent randomly, followed by sequences where the cells are clockwise and counter-clockwise calculated.

As fig. 3 shows, characteristic Moore neighborhood patterns evolve and constantly alternate in steps of $\frac{\pi}{4}$. The change of the sequence from clockwise to counter-clockwise results in a complementing structure ($D^+ \rightarrow D^-$, fig. 3). Areas those before contain diagonal vectors are now contain horizontal/vertical motion direction vectors. Considering the Moore metric (2) the designated paths are equivalent to their walking distance.

Each of the previously created direction fields (D^+ and D^-) represents one component of the final direction field. At the first step the cell based motion direction vectors have to be combined, so sequently aligned vectors are summed up to the point where the direction changes (fig. 4 left). Now each particular cell

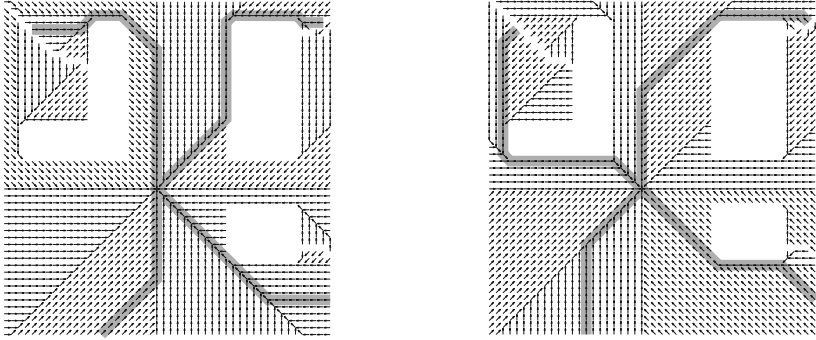


Fig. 3. Different characteristic of simple direction field based on the expansion of the flood fill algorithm. Expanding cells clockwise (D^+ , left) and counter-clockwise (D^- , right).

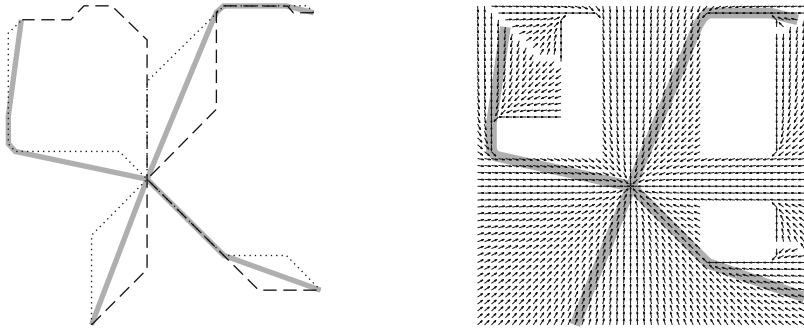


Fig. 4. Combining the particular motion direction vectors from D^+ and D^- results in the precise direction field

contains a diagonal and a horizontal/vertical direction component (fig. 4, right). The final direction field indicates no directional artifacts and the declared paths are consistent to the Euclidean metric.

4 Summary and Outlook

The proposed algorithm efficiently prevents the directional artifacts by combining two different simple direction fields, which are based on Moore metric distance calculation. Due to the fact, that the flood fill algorithm for creating a distance potential is widely used in agent simulation, our enhanced approach provides a fundamental contribution to existing simulation systems. The introduced direction field is an essential part of the route planning component inside the virtual terminal environment [22] of the Institute of Logistics and Aviation.

Future investigations regarding to group dynamic behavior or route planning in the airport terminal environment at normal operations or emergency cases will benefit from our proposed algorithm.

References

- [1] de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry. Springer, Heidelberg (1997) ISBN:3-54061-270-X
- [2] Khatib, O.: The Potential Field Approach and Operational Space Formulation in Robot Control. In: Narendra, K. (ed.) Adaptive and Learning Systems – Theory and Application, pp. 367–377. Plenum Press, New York (1986) ISBN:978-0306422638
- [3] Latombe, J.-C.: Robot Motion Planning, 7th edn. Kluwer Academic Publishers, Dordrecht (1991) ISBN:978-0792391296
- [4] Bruns, H.: Das Eikonale. S. Hirzel (1895)
- [5] Frank, P.: Über die Eikonalgleichung in allgemein anisotropen Medien. *Annalen der Physik* 389(23) (1927)
- [6] Osher, S., Sethian, J.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of computational physics* 79(1), 12–49 (1988)
- [7] Kimmel, R., Sethian, J.: Computing geodesic paths on manifolds. *Proc. Natl. Acad. Sci. USA*, 8431–8435 (1998)
- [8] Sethian, J.: Level Set Methods and Fast Marching Methods. Cambridge University Press, Cambridge (1999) ISBN: 978-0521645577
- [9] Zhao, H.: A fast sweeping method for eikonal equations. *Mathematics of computation* 74(250), 603–628 (2005)
- [10] Jeong, W.-K., Whitaker, R.: A fast eikonal equation solver for parallel systems. In: SIAM Conference on Computational Science and Engineering (2007)
- [11] Gremaud, P., Kuster, C.: Computational study of fast methods for the eikonal equation. *SIAM journal on scientific computing* 27(6), 1803–1816 (2006)
- [12] Kretz, T., Bönnisch, C., Vortisch, P.: Comparison of Various Methods for the Calculation of the Distance Potential Field. In: PED 2008, pp. 335–346. Springer, Heidelberg (2010) ISBN: 978-3-642-04503-5
- [13] Hughes, R.: A continuum theory for the flow of pedestrians. *Transportation Research Part B* 36(6), 507–535 (2002)
- [14] Hoogendoorn, S., Bovy, P.: Pedestrian route-choice and activity scheduling theory and models. *Transportation Research Part B* 38(2), 169–190 (2004)
- [15] Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. In: ACM SIGGRAPH 2006 Papers, p. 1168 (2006)
- [16] Shopf, J., Oat, C., Barczak, J.: GPU Crowd Simulation. In: ACM Transactions on Graphics, Siggraph Asia 2008, vol. 27 (2008)
- [17] Kretz, T.: Pedestrian Traffic: on the Quickest Path. *Journal of Statistical Mechanics: Theory and Experiment* P03012 (2009)
- [18] Bleiweiss, A.: Multi agent navigation on gpu. Eprint (2009)
- [19] Steffen, B., Seyfried, A.: Modeling of pedestrian movement around 90 and 180 degree bends. In: Workshop on Fire Protection and Life Safety in Buildings and Transportation Systems (2009)

- [20] Kretz, T.: Applications of the Dynamic Distance Potential Field Method. In: Dai, S., et al. (eds.) Traffic and Granular Flow 2009 (2010) (in press)
- [21] Lee, C.: An algorithm for path connection and its application. IRE Transactions on Electronic Computers EC-10(3), 346–365 (1961)
- [22] Schultz, M.: Entwicklung eines individuenbasierten Modells zur Abbildung des Bewegungsverhaltens von Passagieren im Flughafenterminal. PhD thesis, Technische Universität Dresden (2010)

Phase Coexistence in Congested States of Pedestrian Dynamics

Armin Seyfried¹, Andrea Portz¹, and Andreas Schadschneider²

¹ Jülich Supercomputing Centre, Forschungszentrum Jülich, 52425 Jülich, Germany
`{a.seyfried,a.portz}@fz-juelich.de`

² Institut für Theoretische Physik, Universität zu Köln, 50937 Köln, Germany
`as@thp.uni-koeln.de`

Abstract. Experimental results for congested pedestrian traffic are presented. For data analysis we apply a method providing measurements on an individual scale. The resulting velocity-density relation shows a coexistence of moving and stopping states revealing the complex structure of pedestrian fundamental diagrams and supporting new insights into the characteristics of pedestrian congestions. Furthermore we introduce a model similar to event driven approaches. The velocity-density relation as well as the phase separation is reproduced. Variation of the parameter distribution indicates that the diversity of pedestrians is crucial for phase separation.

Keywords: crowd dynamics, velocity-density relation.

1 Introduction

The velocity-density relation is one of the most important characteristics for the transport properties of any traffic system. For pedestrian traffic there is currently no consensus even about the principle shape of this relation which is reflected e.g. in conflicting recommendations in various handbooks and guidelines [1]. Discrepancies occur in particular in the high-density regime which is also the most relevant for applications in safety analysis like evacuations or mass events. At high densities stop-and-go waves occur indicating overcrowding and potentially initiating dangerous situations due to stumbling etc. However, the densities where the flow breaks down due to congestion ranges from densities of $\rho_{\max} = 3.8 \text{ m}^{-2}$ to $\rho_{\max} = 10 \text{ m}^{-2}$ [1]. This large variation in values for ρ_{\max} reported in the literature is partly due to insufficient methods of data capturing and data analysis. In previous experimental studies, different kinds of measurement methods are used and often a mixture of time and space averages are realized. Especially in the case of spatial and temporal inhomogeneities the choice of the measurement method and the type of averaging have a substantial influence on the results [2].

Up to now, congested states in pedestrian dynamics have not been analyzed in much detail. This is in contrast to vehicular traffic where the congested phase is well-investigated, both empirically and theoretically [3,4].

In this contribution we show that even improved classical measurement methods using high precision trajectories but basing on mean values of density and velocity fail to resolve important characteristics of congested states. For a thorough analysis of pedestrian congestion we apply a new method enabling measurements on the scale of single pedestrians.

2 Experimental Data

For our investigation we use data from experiments performed in 2006 in the wardroom of Bergische Kaserne Düsseldorf with a test group of up to $N = 70$ soldiers. The length of the circular system was about 26 m, with a $l = 4$ m long measurement section. Detailed information about the experimental setup and data capturing providing trajectories of high accuracy ($|x_{\text{err}}| \leq 0.02$ m) is given in [25].

In Fig. 1 the x -component of trajectories is plotted against time. For the extraction of the trajectories, the pedestrians' heads were marked and tracked.

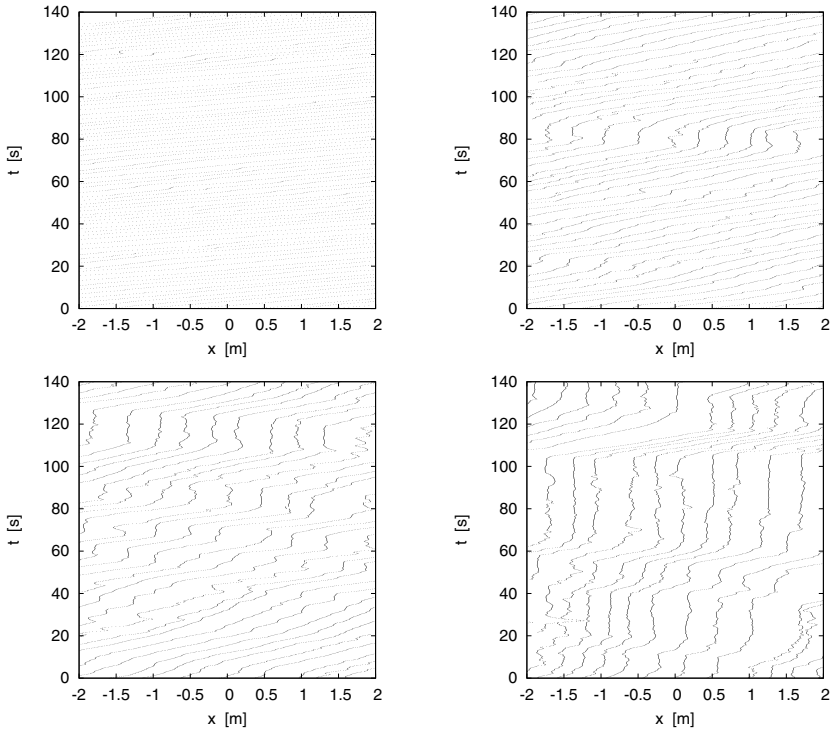


Fig. 1. Trajectories for the runs with $N = 39, 56, 62$ and 70 (left to right, top to bottom). With increasing density the occurrence of stop-and-go waves accumulate.

Backward movement leading to negative velocities is caused by head movement of the pedestrians during a standstill. Inhomogeneities in the trajectories increase with increasing density. As in vehicular traffic, jam waves propagating opposite to the movement direction (upstream) occur at higher densities.

Stopping is first observed during the runs with $N = 45$ pedestrians, at 70 pedestrians they can hardly move forward. Macroscopically one observes separation into a stopping area and an area where pedestrians walk slowly. In the following we analyze how macroscopic measurements blur this phase separation and apply a technique introduced in [6] enabling a measurement of the fundamental diagram on a ‘microscopic’ scale.

2.1 Macroscopic Measurement

Speed v_i of pedestrian i and the associated density ρ_i are calculated using the entrance and exit times t_i^{in} and t_i^{out} into and out of a measurement section of length $l_m = 2$ m,

$$v_i = \frac{l_m}{t_i^{\text{out}} - t_i^{\text{in}}}, \quad \rho_i = \frac{1}{t_i^{\text{out}} - t_i^{\text{in}}} \int_{t_i^{\text{in}}}^{t_i^{\text{out}}} \rho(t) dt \quad \text{with} \quad \rho(t) = \frac{\sum \Theta_i(t)}{l_m}. \quad (1)$$

The speed v_i is a mean value over the space-time interval $\Delta t_i = t_i^{\text{out}} - t_i^{\text{in}}$ and $\Delta x = l_m$. By integration over the instantaneous density $\rho(t)$ the density is assigned to the same space-time interval. To reduce the fluctuations of $\rho(t)$ we use the quantity $\Theta_i(t)$, introduced in [7], which measures the fraction of space between pedestrians i and $i + 1$ that is inside the measurement area.

Results of the macroscopic measurement method are shown in Fig. 2. In comparison to method B introduced in [2] (see Fig. 6 of [2] which uses data based

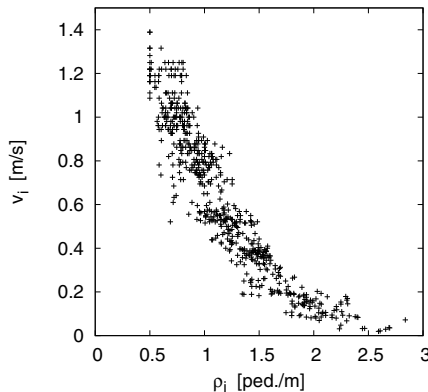


Fig. 2. Velocity-density relation using an improved macroscopic measurement method. There is no indication of phase separation since stopping states ($v \approx 0$) occur only at large densities where no moving states ($v > 0$) are observable.

on the same trajectories as this study) the scatter of the data is reduced due to an improved density definition and a better assignment of density and velocity. However the resulting velocity-density relation does not allow to identify phase-separated states although these are clearly visible in the trajectories.

2.2 Microscopic Measurement

To identify phase separated states we determine the velocity-density relation on the scale of single pedestrians. This can be achieved by the Voronoi density method [6]. In one dimension a Voronoi cell is bounded by the midpoints $z_i = (x_{i+1} + x_i)/2$ of the pedestrian positions x_i and x_{i+1} . With the length $L_i = z_i - z_{i-1}$ of the Voronoi cell corresponding to pedestrian i and $\Delta t = 0.5$ s we define the instantaneous velocity and density by

$$v'_i(t) = \frac{x_i(t + \Delta t/2) - x_i(t - \Delta t/2)}{\Delta t}, \quad \rho'_i(x) = \begin{cases} 1/L_i & : x \in [z_i, z_{i+1}[\\ 0 & : \text{otherwise} \end{cases} . \quad (2)$$

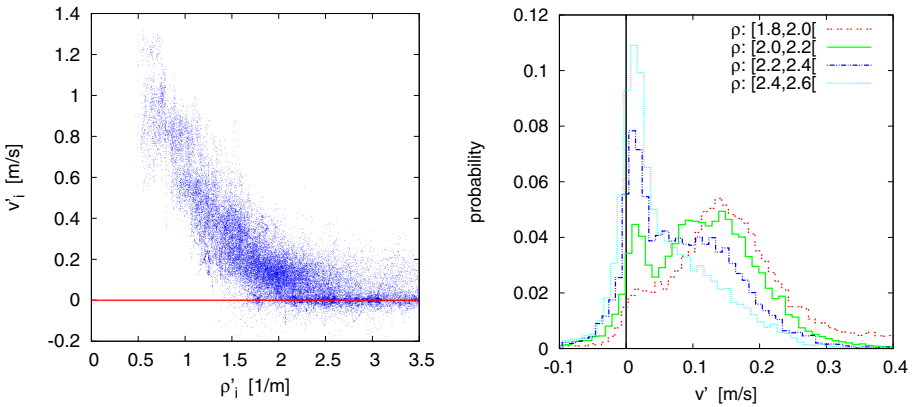


Fig. 3. Left: Velocity-density relation on an individual scale. At each density $\rho' > 1.5 \text{ m}^{-1}$ both stopping and moving states are observable. **Right:** Probability distribution of the velocities for different density intervals. The double peak structure indicates the coexistence of moving and stopping states. The height of the stopping peak increases with increasing density.

The fundamental diagram based on the Voronoi method is shown on the left side of Fig. 3. Regular stops occur at densities higher than 1.5 m^{-1} . On the right side of Fig. 3 the distribution of the velocities for fixed densities from 1.8 m^{-1} to 2.6 m^{-1} are shown. There is a continuous change from a single peak near $v = 0.15 \text{ m/s}$, to two peaks, to a single peak near $v = 0 \text{ m/s}$. The right peak represents the moving phase, whereas the left peak represents the stopping phase. At densities around 2.2 m^{-1} these peaks coexist, indicating phase separation into a flowing and a jammed phase.

2.3 Phase Separation in Vehicular Traffic

In highway traffic, phase separation into moving and stopping phases typically occurs when the outflow from a jam is reduced compared to maximal possible flow in the system. Related phenomena are hysteresis and a non-unique fundamental diagram. At intermediate densities two different flow values can be realized. The larger flow, corresponding to a homogeneous state, is metastable and breaks down due to fluctuations or perturbations (capacity drop). The origin of the reduced jam outflow is usually ascribed to the so-called slow-to-start behaviour (see [34] and references therein), i.e. an delayed acceleration of stopped vehicles due to the loss of attention of the drivers etc.

The structure of the phase-separated states in vehicular traffic is different from the ones observed here. For vehicle traffic the stopping phase corresponds to a jam of maximal density whereas in the moving phase the flow corresponds to the maximal *stable* flow, i.e. all vehicles in the moving phase move at their desired speed. This scenario is density-independent as increasing the global density will only increase the length of the stopping region without reducing the average velocity in the free flow regime. The probability distribution of the velocities (in a periodic system) shows a similar behaviour to that observed in Fig. 3. The position of the free flow peak in the case of vehicular traffic is independent of the density.

The behaviour observed here for pedestrian dynamics differs slightly from that described above. The main difference concerns the properties of the moving regime. Here the observed average velocities are much smaller than the free walking speeds. Therefore the two regimes observed in the phase separated state are better characterized as “stopping” and “slow moving” regimes.

Further empirical studies are necessary to clarify the origin of these differences. One possible reason are the different acceleration properties of vehicles and pedestrians as well as anticipation effects. It also remains to be seen whether in pedestrian systems phenomena like hysteresis can be observed.

3 Modeling

3.1 Adaptive Velocity Model

In this section we introduce the adaptive velocity model, which is based on an event driven approach [9]. A pedestrian can be in different states which determine the velocity. A change between these states is called *event*. The model was derived from force-based models, where the dynamics of pedestrians are given by the following system of coupled differential equations

$$m_i \frac{dv_i}{dt} = F_i \quad \text{with} \quad F_i = F_i^{\text{drv}} + F_i^{\text{rep}} \quad \text{and} \quad \frac{dx_i}{dt} = v_i, \quad (3)$$

where F_i is the force acting on pedestrian i . The mass is denoted by m_i , the velocity by v_i and the current position by x_i . F_i is split into a repulsive force F_i^{rep} and a driving force F_i^{drv} . The dynamics is regulated by the interrelation

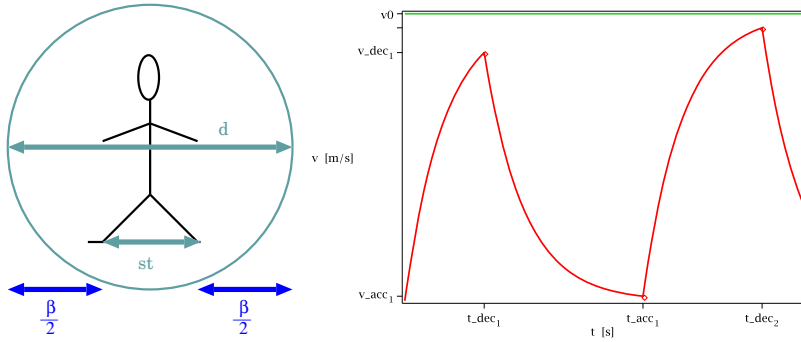


Fig. 4. Left: Connection between the required space d , the step length st and the safety distance β . **Right:** The adaptive velocity: acceleration until t_{dec1} than deceleration until t_{acc1} , again acceleration until t_{dec2} and so on.

between driving and repulsive forces. In our approach the role of repulsive forces are replaced by events. The driving force is defined as

$$F_i^{drv} = \frac{v_i^0 - v_i}{\tau_i}, \tag{4}$$

where v_i^0 is the desired speed of a pedestrian and τ the relaxation time of the velocity. By solving the differential equation

$$\frac{dv_i}{dt} = F_i^{drv} \Rightarrow v_i(t) = v_i^0 + c \exp\left(-\frac{t}{\tau_i}\right), \quad \text{with } c \in \mathbb{R}, \tag{5}$$

the velocity function is obtained. This is shown in Fig. 4 together with the parameters governing the pedestrians' movement. In this model pedestrians are treated as bodies with diameter d [9]. The diameter depends linearly on the current velocity and is equal to the step length st in addition to the safety distance β

$$d_i(t) = st_i(t) + \beta_i(t) \tag{6}$$

Step length and safety distance are introduced to define the rules for the dynamics of the system. We determine the model parameters from empirical data which allows to judge the adequacy of the rules. Based on [10] the step length is a linear function of the current velocity with following parameters:

$$st_i(t) = 0.235m + 0.302[s] v_i(t). \tag{7}$$

The required quantities for the safety distance can be specified through empirical data of the fundamental diagram $\bar{d}_i = 1/\rho = 0.36 + 1.06 v$, see [7]. With these experimental results the previous equations can be summarized to

$$\beta_i(t) = d_i(t) - st_i(t) = a_i + b_i v_i(t) \tag{8}$$

with $a_i = 0.125$ m and $b_i = 0.758$ s. No free model parameter remain with these specifications. In the following we describe the rules for the movement. A pedestrian accelerates to the desired velocity v_i^0 until the distance $\Delta x_{i,i+1}$ to the pedestrian in front is smaller than the safety distance. From this time on, he/she decelerates until the distance is larger than the safety distance. To guarantee a minimal volume exclusion, case “collision” is included, in which the pedestrians are too close to each other and have to stop. Via $\Delta x_{i,i+1}$, d_i and β_i the velocity function for the states deceleration (dec.), acceleration (acc.) and collision (coll.) can be defined, see Eq. 9:

$$v_i(t) = \begin{cases} v_{\text{dec}} \exp\left(-\frac{t-t_{\text{dec}}}{\tau}\right) & \text{for } \Delta x_{i,i+1} - \delta_i(t) \leq 0 & \text{(dec.)} \\ v_i^0(1 - \exp\left(-\frac{t-t_0}{\tau}\right)) & \text{for } \Delta x_{i,i+1} - \delta_i(t) > 0 & \text{(acc.)} \\ 0 & \text{for } \Delta x_{i,i+1} - \delta_i(t) \leq -\beta_i(t)/2 & \text{(coll.)} \end{cases} \quad (9)$$

where $\delta_i(t) = (d_i(t) + d_{i+1}(t))/2$ is the distance between the centers of both pedestrians. The current velocity $v_i(t)$ of a pedestrian i depends on his/her state. t_{dec} denominates the point in time where a change from acceleration to deceleration takes place. Conversely t_{acc} is the change from deceleration to acceleration. $v_{\text{dec}} = v(t_{\text{dec}})$ and $v_{\text{acc}} = v(t_{\text{acc}})$ are defined accordingly. At the beginning $t_0 = 0$ s with a change from acceleration to deceleration a new calculation of t_0 is necessary:

$$t_0 = t_{\text{acc}} + \ln\left(1 - \frac{v_{\text{acc}}}{v_0} \exp\left(\frac{-t_{\text{acc}} - t_{\text{dec}}}{\tau}\right)\right) \quad (10)$$

The discreteness of the time step could lead to configurations where overlapping occurs. To ensure good computational performance for high densities, no events are explicitly calculated. Instead in each time step, it is checked whether an event has taken place and t_{dec} , t_{acc} or t_{coll} are set to t accordingly. To avoid too large interpenetration of pedestrians and to implement a reaction time in a realistic size we choose $\Delta t = 0.05$ s. To guarantee a parallel update a recursive procedure is necessary: Each person is advanced one time step according to Eq. 9. If after this step a pedestrian is in a different state because of the new distance to the pedestrian in front, the velocity is set according to this state. Then the state of the next following person is reexamined. If the state is still valid the update is completed. Otherwise, the velocity is calculated again.

3.2 Model Validation and Influence of Individual Differences

In the following we face model results with experimental data and study how the distribution of individual parameter influences the phase separation. For all simulations the desired velocity is normal distributed $v_i^0 \sim \mathcal{N}(\mu, \sigma^2)$ with average $\mu = 1.24$ m/s and variance $\sigma^2 = 0.05$. Fig. 5 and Fig. 6 show the simulation results for two different choices of parameter distributions.

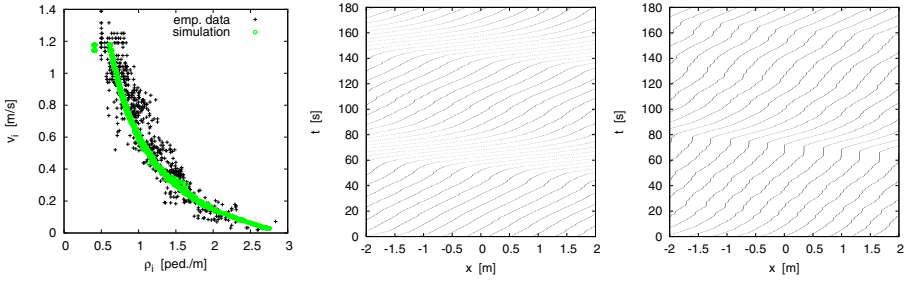


Fig. 5. Validation of the modeled fundamental diagram and trajectories with same parameter a_i , b_i and τ_i for all pedestrians **Left:** Comparison of fundamental diagrams of modeled and empirical data. **Middle:** Trajectories for $N = 62$ (model). **Right:** Trajectories for $N = 70$ (model).

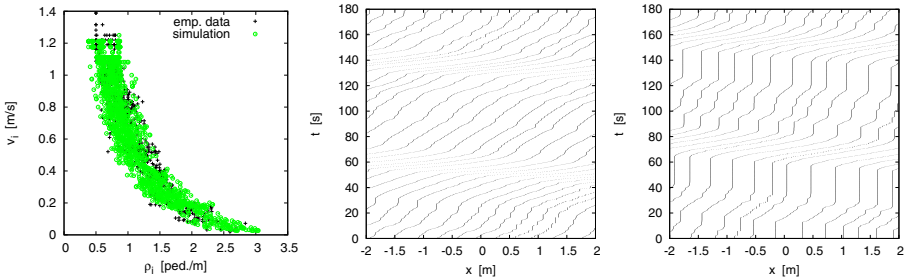


Fig. 6. Validation of the modeled fundamental diagram and trajectories, with normal distributed individual parameter. **Left:** Comparison of fundamental diagrams of modeled and empirical data. **Middle:** Trajectories for $N = 62$ (model). **Right:** Trajectories for $N = 70$ (model).

The model yields the right macroscopic relation between velocity and density even if a_i , b_i , st_i and τ_i are the same for all pedestrians, see Fig. 5 (left). The trajectories display that phase separation does not appear. Even at high densities the movement is ordered and no stops occur. For further simulations we incorporate a certain disorder by choosing the following individual parameter normal distributed: $a_i \sim \mathcal{N}(0.125, 0.1)$, $b_i \sim \mathcal{N}(0.758, 0.5)$ and $\tau_i \sim \mathcal{N}(1.0, 0.1)$.

Variation of the personal parameters affects the scatter of the fundamental diagram, see Fig. 6 left. Phase separation appears in the modeled trajectories as in the experiment, see Fig. 6 middle and right. It is clearly visible that long stop phases occur by introducing distributed individual parameters. Then the pattern as well as the change of the pattern from $N = 62$ to $N = 70$ are in good agreement with the experimental results, see Fig. 1. Even the phase separated regimes match qualitatively. However, the regimes appear more regular in the modeled trajectories.

In Fig. 7 microscopic measurements of fundamental diagram and the related velocity distributions are shown. Separation of phases is reproduced well. But the position of the peak attributed to the moving phase is not in conformance with the experimental data, compare Fig. 3 (right). The experimental data show that the peak position is independent from the density at v around 0.15 m/s. At the model data the position of the peak changes with increasing density. Measurements with different time steps show that the size of the time step influences the length and shape of the stop phase at high densities. But the density where first stops occur seems independent from the size of the time step. Further model analysis is necessary to study the role of the reaction time implemented by discrete time steps in this special type of update. Furthermore we will study how the change of the peak could be influenced by including a distribution for the step length and other variations of the distribution for the safety distance.

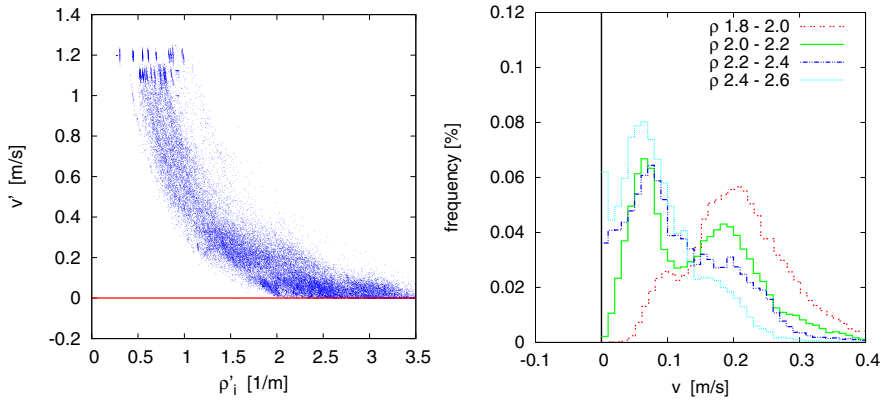


Fig. 7. Left: Microscopic fundamental diagram of the modeled data with normal distributed individual parameters. **Right:** Distribution of v at fixed densities.

4 Conclusion

We have investigated the congested regime of pedestrian traffic using high-quality empirical data based on individual trajectories. Strong evidence for phase separation into standing and slow moving regimes is found. The corresponding velocity distributions show a typical two-peak structure. The structure of the trajectories is well reproduced by an adaptive velocity model which is a variant of force-based models in continuous space.

Future studies should clarify the origin of the differences to the phase separated states observed in vehicular traffic. Here phase separation into a stopping and a moving phase occurs such that the average velocity in the moving regime is independent of the total density.

Acknowledgments. This work is supported by the Federal Ministry of Education and Research - BMBF (FKZ 13N9952 and 13N9960) and by the German Research Foundation - DFG (KL 1873/1-1 and SE 1789/1-1).

References

1. Schadschneider, A., Klingsch, W., Klüpfel, H., Kretz, T., Rogsch, C., Seyfried, A.: Evacuation Dynamics: Empirical Results, Modeling and Applications. In: Meyers, R.A. (ed.) *Encyclopedia of Complexity and System Science*, pp. 3142–3176. Springer, Heidelberg (2009)
2. Seyfried, A., Boltes, M., Kähler, J., Klingsch, W., Portz, A., Schadschneider, A., Steffen, B., Winkens, A.: Enhanced empirical data for the fundamental diagram and the flow through bottlenecks. In: Klingsch, W.W.F., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) *Pedestrian and Evacuation Dynamics 2008*, pp. 145–156. Springer, Heidelberg (2010)
3. Chowdhury, D., Santen, L., Schadschneider, A.: Statistical physics of vehicular traffic and some related systems. *Phys. Rep.* 329, 199 (2000)
4. Schadschneider, A., Chowdhury, D., Nishinari, K.: *Stochastic Transport in Complex Systems*. Elsevier, Amsterdam (2010)
5. Boltes, M., Seyfried, A., Steffen, B., Schadschneider, A.: Automatic Extraction of Pedestrian Trajectories from Video Recordings. In: Klingsch, W.W.F., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) *Pedestrian and Evacuation Dynamics 2008*, pp. 43–54. Springer, Heidelberg (2010)
6. Steffen, B., Seyfried, A.: Methods for measuring pedestrians density, flow, speed and direction with minimal scatter. *Physica A* 389, 1902–1910 (2010)
7. Seyfried, A., Steffen, B., Klingsch, W., Boltes, M.: The fundamental diagram of pedestrian movement revisited. *J. Stat. Mech.* P10002 (2005)
8. Portz, A., Seyfried, A.: Analyzing Stop-and-Go Waves by Experiment and Modeling. In: *Pedestrian and Evacuation Dynamics 2010* (to appear, 2010) (Preprint), arxiv:1003.5446v1
9. Chraïbi, M., Seyfried, A.: Pedestrian Dynamics With Event-driven Simulation. In: *Pedestrian and Evacuation Dynamics 2008*, pp. 713–718. Springer, Heidelberg (2010)
10. Weidmann, U.: *Transporttechnik der Fussgänger*. Schriftenreihe des IVT Nr. 90, Institut für Verkehrsplanung, Transporttechnik, Strassen- und Eisenbahnbau. ETH, Zürich (1993)

Stochastic Transition Model for Discrete Agent Movements

Michael Schultz and Hartmut Fricke

Institute of Logistics and Aviation, Technische Universität Dresden,
Hettner Str. 1-3, D-01062 Dresden, Germany
{schultz,fricke}@ifl.tu-dresden.de

Abstract. We propose a calibrated two-dimensional cellular automaton model to simulate pedestrian motion behavior. It is a $v_{max} = 4$ (3) model with exclusion statistics and random shuffled dynamics. The underlying regular grid structure results in a direction-dependent behavior, which has in particular not been considered within previous approaches. We efficiently compensate these grid-caused deficiencies on model level.

Keywords: multi-agent, transition matrix, stochastic approach, calibrated model.

1 Introduction

The different model approaches for microscopic person dynamics are based on the particular discipline analogies, ranging from hydro-dynamic models to artificial intelligence and multi-agent systems [1]. The complex dynamic human behavior is induced by individual decisions, which are classified to be of short-range (operational) and long-range type (strategic/tactical). The self-organization of persons is a further essential characteristic of human behavior. In contrast to the social force model [2-4] or the discrete choice model [5, 6] the developed motion model [7-9] is based on a stochastic approach to handle the unpredictable behavior by individual path deviations. The stochastic motion model is an appropriate and fast method for analysis the dynamic pedestrian behavior. However, to derive valid results several simulation runs (>100) have to be performed. The focus concentrates on the evaluation of application oriented simulation scenarios instead of the characteristics of individual interactions or specific pedestrian trajectories.

2 Stochastic Motion Model

The presented motion model is based on a stochastic approach [10], which is comparable to a common cellular automaton. It utilizes a regular grid structure. In contrast to the cellular automaton, the new model is developed on the basis of a fundamental paradigm shift: instead of changing the cell status depending on the status of its surrounding cells (neighbors), the agent is able to move

over the regular lattice and to enter those cells, which are not occupied by other agents or obstacles (e.g. walls). To describe the motion behavior of an agent, the motion vector is separated into a desired motion direction and a transversal deviation [10]. Using the spatially discrete grid structure and defining three transition states (forward | stop | backward or left | on track | right) the normalized transition probability (p) into these states is generally defined by the following equations.

$$\begin{aligned}
 p^+ &= \frac{1}{2} (\sigma^2 + \mu^2 + \mu) \quad , \text{for } \textit{forward} \text{ or } \textit{left} \\
 p^o &= 1 - (\sigma^2 + \mu^2) \quad , \text{for } \textit{stop} \text{ or } \textit{on track} \\
 p^- &= \frac{1}{2} (\sigma^2 + \mu^2 - \mu) \quad , \text{for } \textit{backward} \text{ or } \textit{right}
 \end{aligned}
 \tag{1}$$

In the case of the desired motion direction, μ denotes the desired speed and σ^2 the corresponding variance. If the transversal deviation is concerned, μ is the average and σ^2 is the range of the fluctuations. Considering a symmetric transversal deviation ($\mu_{\text{deviation}} = 0$) and a connection of desired speed and the corresponding variance (no step backward $p^- = 0$, so that $\sigma_{\text{speed}}^2 = \mu_{\text{speed}}(1 - \mu_{\text{speed}})$), the above equations are simplified to the following equations for the desired motion direction (2) and for the transversal motion direction (3).

$$p^{\text{forward}} = \mu_{\text{speed}} \quad | \quad p^{\text{stop}} = 1 - \mu_{\text{speed}}
 \tag{2}$$

$$p^{\text{left,right}} = \frac{1}{2} \sigma_{\text{deviation}}^2 \quad | \quad p^{\text{on track}} = 1 - \sigma_{\text{deviation}}^2
 \tag{3}$$

Finally, the motion components are combined to a 3x3 transition matrix (M_{ij}) as shown in the following fig. 1. The emphasized cell (marked gray at the figure) contains the transition probability of moving forward without transversal deviations. In fact, the transition matrix possesses a two-dimensional characteristic, but it only defines an one-dimensional transition considering a transversal deviation (1.5-dimensional).

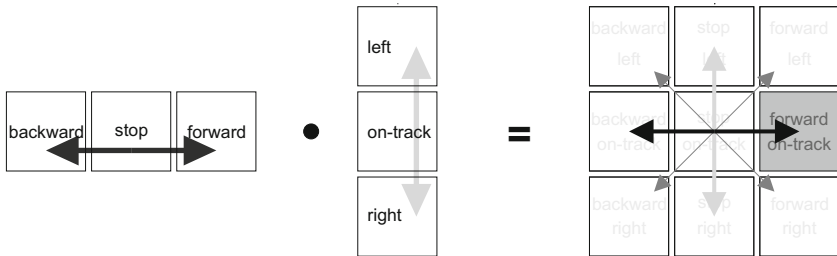


Fig. 1. Generation of the transition matrix due to combination of desired motion speed and transversal motion deviation

To allow for a three-dimensional agent motion behavior, two independent motion directions are needed. Based on the developed horizontal transition matrix (\hat{M}) a diagonal transition matrix (\tilde{M}) is derived by re-indexing the horizontal matrix (turning the matrix by $\frac{\pi}{4}$, [10]). The motion direction (α) is integrated into the stochastic model by superpose these matrices with λ .

$$M = \begin{cases} (1 - \lambda) \hat{M} + \lambda \tilde{M}, & \lambda = \tan \alpha, 0 \leq \alpha < \frac{\pi}{4} \\ \frac{1}{\sqrt{2}} (1 - \lambda) \hat{M} + \sqrt{2} \lambda \tilde{M}, & \lambda = \tan (\frac{\pi}{4} - \alpha), \frac{\pi}{4} \leq \alpha \leq \frac{\pi}{2} \end{cases} \tag{4}$$

The rotation of M (4-fold symmetry) allows for determining the entire spectrum of the motion direction. The underlying regular grid structure results in a direction dependent behavior (e.g. entering diagonal cells implies walking a longer way in comparison to horizontally located cells). Therefore the first model modification is to adjust λ within the parameters of $\frac{\pi}{4} \leq \alpha \leq \frac{\pi}{2}$.

It's obvious, that the stochastic motion model allows for horizontal and diagonal movements (fig. 2, Moore-Neighborhood).

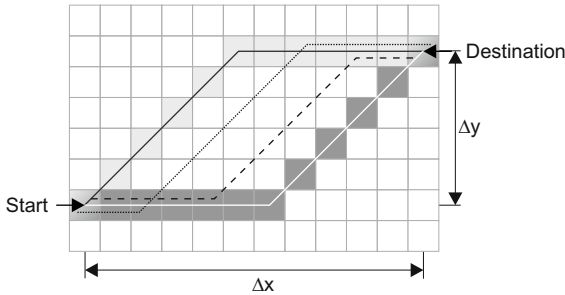


Fig. 2. The distance metric d results from the Moore-Neighborhood

The shortest distance between two points (start, destination) is given by the metric d (5) which differs from the defined p-norms (Manhattan, Euclidean and Chebychev norm).

$$d = |\Delta x - \Delta y| + \sqrt{2} \min (\Delta x, \Delta y) \tag{5}$$

3 Model Constraints, Improvements and Validation

Due to the utilization of a regular grid structure, the transition matrix does not fulfill the criteria of independent agent motion behavior. So, the speed and the variance of the agent depend on the agent motion direction. If the agent enters diagonal cells his walking distance is longer (approx. 41%) in comparison to the use of horizontally located cells. This model constraint is equivalent to a

significant higher motion speed depending on the direction of motion. Algorithms to compensate this grid-based speed effect can be found at [7, 11]. A detailed model analysis points out that the expected value of the transition matrix μ_M (6), defined by cell based transition probability M_{ij} and the relative location \mathbf{e}_{ij} , differs from μ_{speed} , which is specified in the motion model.

$$\mu_M = \sum_i \sum_j \mathbf{e}_{ij} M_{ij} \quad , \quad \text{with } \mathbf{e}_{ij} = \begin{pmatrix} i \\ j \end{pmatrix} \quad (6)$$

Fig. 3 points out the correlation of speed and motion angle. With increasing α the corresponding speed (μ_{speed}) increases as well, whereas the stepwise change of the transversal deviation ($\sigma_{\text{deviation}}^2$) mitigates the α -depending characteristics.

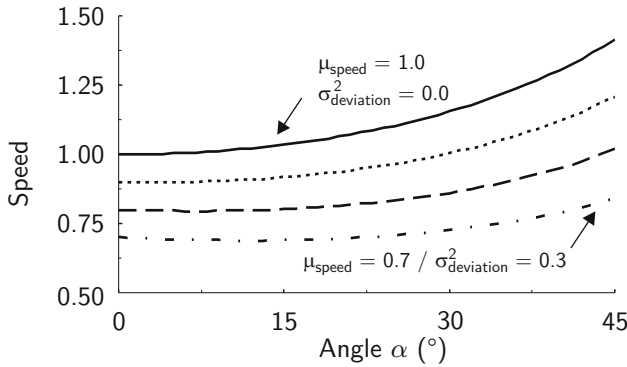


Fig. 3. Model deficiency due to motion angle α depending characteristic of speed

The stochastic motion model allows for absolute cell transitions ($M_{ij} = 1$ with $\sigma_{\text{deviation}}^2 = 0$) only at motion angle of $\alpha = 0$ and $\alpha = \frac{\pi}{4}$. If the agents choose another angle he always has to choose between two cells at least. Because of the superposition of the horizontal (\hat{M}) and diagonal (\tilde{M}) transition matrices, a model immanent variance from desired agent motion direction occurs, even if the model defines $\sigma_{\text{deviation}}^2 = 0$). The characteristic of this overall motion model variance is shown in fig. 4

The angle depended variance of motion implies to major issues. The model shows an immanent motion deviation, which is not considered in previous equations and the different variances lead to different avoiding behavior. Using the parameter set $\mu_{\text{speed}} = 1$ and $\sigma_{\text{deviation}}^2 = 0$ as an example, at $\alpha = 0$ no variance is allowed by the model and an agent cannot move if the chosen cell is blocked. Using the same scenario with a different motion angle (e.g. $\alpha = \frac{\pi}{8}$), the agent gets the probability of approx. 20 % to pass the blocked cell. To ensure homogeneous variance, an appropriate compensation on model level is needed. The expected value μ_M of the matrix depends on μ_{speed} and $\sigma_{\text{deviation}}^2$ whereas the parameters

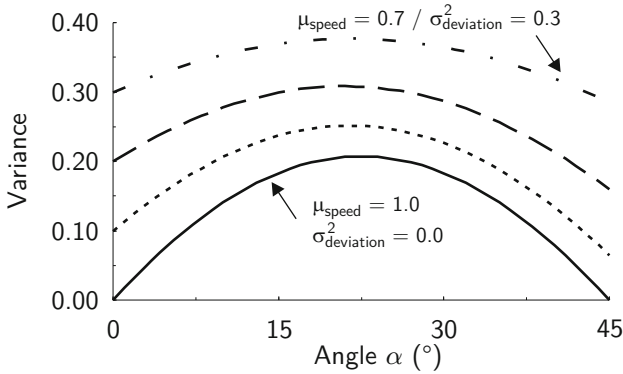


Fig. 4. Model deficiency due to motion angle depending characteristic of variance

are directly coupled. For each parameter set a specific characteristic over the angle has to be calculated. The following fig. 5 shows these characteristics.

Further model investigations point out that the model has to be extended to reproduce the representative shape of the fundamental diagram [12, 13]. Using the transition matrix, an agent is able to react to the status of the adjacent cells (empty, occupied). The fundamental diagram indicates an interaction range of about 1.3 m, because the speed of an agent starts to decrease if the density relations ρ/ρ_{max} reaches a level of 10 % (considering an agent with a dimension of 0.4×0.4 m and a maximum density of $\rho_{max} = 6.25$ Person/m²). If the agent moves three/ four steps at once, he will be able to interact with distant agent

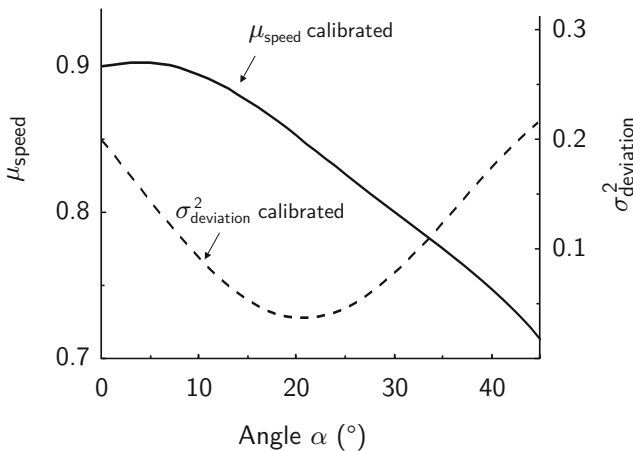


Fig. 5. Functional characteristics of the parameter compensation for the parameter set $\mu_{speed} = 0.9$ and $\sigma_{deviation}^2 = 0.2$

and the developed model is found to reproduce the characteristic shape of the fundamental diagram (fig. 6). Therefore the motion model has to provide the following agent properties:

- Always move, no waiting (occupied cells increase transition probability of the other matrix cells).
- Move four steps at once and decrease the steps depending on agent density, at least in the case of $\rho/\rho_{\max} > 0.6$ the number of steps should be reduced from four to three to fit the shape.
- Agent leaves a trace, at each time step all entered cells will temporarily blocked.

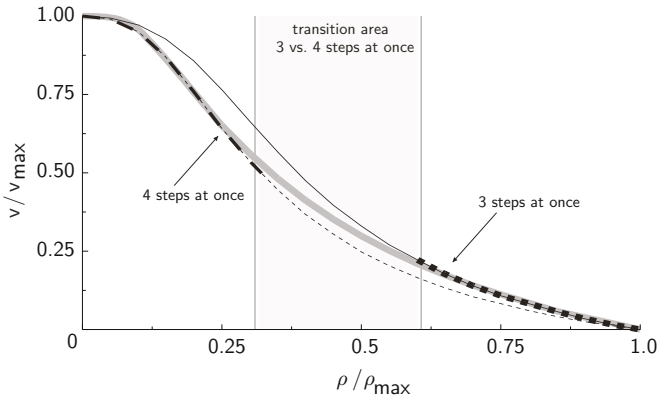


Fig. 6. Fundamental diagram of stochastic motion model using different agent operating distances (3-4 steps at once) depending on agent density

The stochastic model meets all criteria for a scientifically reliable motion model. It exhibits the absence of significant model-caused limitations and reproduces all common self-organizing effects (e.g. row formation or oscillation). Besides the operational motion definition by the stochastic transition matrix, strategic/tactical motion components are taken into account as well. The stochastic model allows for the reaction of the agent to objects/agents at immediate vicinity. It additionally provides the capability of considering distant constellation of agents (jam) and potentially blocked bottlenecks.

4 Summary and Outlook

Model specific parameter corrections ensure that the motion vector is equal to the expected value of the corresponding transition matrix. This issue has in particular not been considered within previous approaches. The calibrated motion model is thus the first approach, which allows for a specific stochastic description of agent movements without model restrictions.

The passenger related evaluations and simulations of dispatch processes at Dresden Airport exemplarily show that the developed stochastic motion model is able to reproduce the behavior of passengers in an appropriate way [8, 9]. Therefore the developed motion model is implemented in a corresponding application environment which allows for a various application, e.g. investigations regarding to group dynamic behavior or route planning in the airport terminal focused on normal operations and emergency cases.

References

- [1] Schadschneidern, A.: Evacuation Dynamics: Empirical Results, Modeling and Applications, pp. 3142–3176. Springer, Heidelberg (2009)
- [2] Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. *Phys. Rev. E* 51, 4282–4286 (1995)
- [3] Johansson, A., Helbing, D., Shukla, P.: Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in Complex Systems* 10(4), 271–288 (2007)
- [4] Moussaïd, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLoS One* 5(4) (2010)
- [5] Bierlaire, M., Antonini, G., Weber, M.: Behavioral dynamics for pedestrians. In: *Proceedings of the 10th International Conference on Travel Behavior Research*, Lucerne (2003)
- [6] Robin, T., Antonini, G., Bierlaire, M., Cruz, J.: Specification, estimation and validation of a pedestrian walking behavior model. *Transportation Research Part B: Methodological* 43(1), 36–56 (2009)
- [7] Schultz, M., Lehmann, S., Fricke, H.: A discrete microscopic model for pedestrian dynamics to manage emergency situations in airport terminals. In: *Pedestrian and Evacuation Dynamics 2005*, pp. 369–375. Springer, Berlin (2007)
- [8] Schultz, M., Schulz, C., Fricke, H.: Passenger Dynamics at Airport Terminal Environment. In: *Pedestrian and Evacuation Dynamics 2008*, pp. 381–396. Springer, Berlin (2010)
- [9] Schultz, M.: Entwicklung eines individuenbasierten Modells zur Abbildung des Bewegungsverhaltens von Passagieren im Flughafenterminal. PhD thesis, Technische Universität Dresden (2010)
- [10] Burstedde, C., Klauck, K., Schadschneider, A., Zittartz, J.: Simulation of pedestrian dynamics using a 2-dimensional cellular automaton. *Physica A* 295, 507–525 (2001)
- [11] Kretz, T., Schreckenberg, M.: Moore and more and symmetry. In: *Pedestrian and Evacuation Dynamics 2005*, pp. 297–308. Springer, Berlin (2007)
- [12] Weidmann, U.: *Transporttechnik der Fußgänger*. Schriftenreihe des Institut für Verkehrsplanung, Transporttechnik, Strassen- und Eisenbahnbau 90 (1992)
- [13] Seyfried, A., Steffen, B., Klingsch, W., Boltes, M.: The fundamental diagram of pedestrian movement revisited. *J. Stat. Mech.*, P10002 (2005)

Analysis of Obstacle Density Effect on Pedestrian Congestion Based on a One-Dimensional Cellular Automata

Kenichiro Shimura¹, Yuki Tanaka², and Katsuhiro Nishinari¹

¹ Research Center for Advanced Science and Technology, The University of Tokyo
4-6-1, Komaba, Meguro-ku, Tokyo, 153-8904, Japan

² Department of Aeronautics and Astronautics, Graduate School of Engineering
The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
shimura@tokai.t.u-tokyo.ac.jp

Abstract. This is a study on a situation where pedestrians walk through a crowded area. The 1-D Cellular Automata model ultimately resulting to the Asymmetric Simple Exclusion Process is studied. Numerical and analytical study is carried out to investigate how pedestrian behaviour is affected.

Keywords: Cellular Automata, ASEP, Pedestrian, Crowd, Burgers' equation.

1 Introduction

Congestion always appeared to be a great social problem over the course of recorded history. Although pedestrian dynamics has been studied through physics [1], it is important to carry out the study with a consideration of a more realistic situation in mind, as the experience of “walking through a crowd” cannot be avoided for individuals living in the city. This study focuses on such a phenomenon, proposing a Cellular Automata (CA) model to describe a situation where pedestrians walk through a crowded area to investigate how crowds may interfere with pedestrian behaviour.

2 Modelling

This is a study on a situation where pedestrians walk through a crowded area. In this model, a crowded area is expressed as the area where obstacles are randomly moving and pedestrians walk through such a crowded area in a straight line. The model ultimately applies the Asymmetric Simple Exclusion Process (ASEP) with additional white noise.

The schematic diagram of this situation is illustrated in Fig. 1a. Each of these cells in the lattice can exclusively contain either a “Pedestrian” or an “Obstacle”. The “Empty” cell contains neither a Pedestrian nor an Obstacle. Here the crowd is considered as a set of Obstacles behaving under the random walk principle. The diagram illustrates the situation where Pedestrians enter the crowded area and walk through in a straight line, coming out of the exit. The dynamics of the Obstacles are

assumed as random walk with the speed faster than Pedestrians. When the number of random walk takes place fast enough during the update interval of the Pedestrians' dynamics, it is assumed that the position of the Obstacles are fairly random at each given time. Thus, it is reasonable to simplify the dynamics of the Obstacles into a simple random replacement. This means that the position of the Obstacles is randomly reshuffled for each time step.

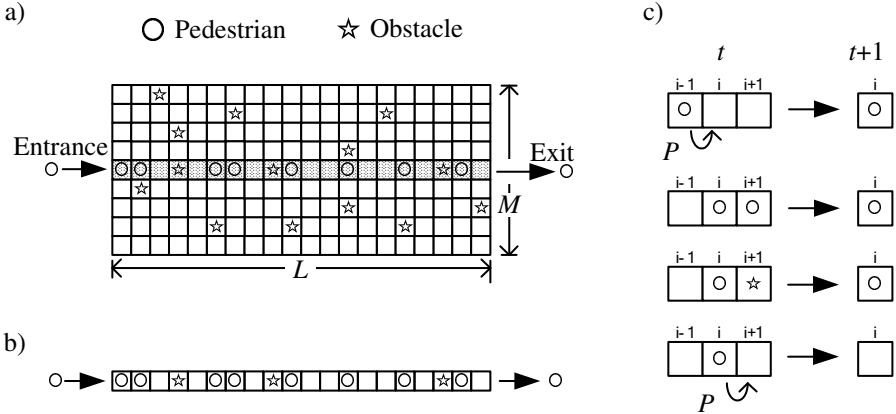


Fig. 1. Model description. a) Illustrates pedestrians passing through the crowded area. Where the open star and open circle denotes the Obstacles and Pedestrians, respectively. L and M denote the number of cells in lateral and vertical directions. The highlighted row illustrates the pedestrian's passage. b) Illustrates the reduced obstacle-pedestrian system in one-dimension. c) Illustrates the transition rules for the reduced one-dimensional model. i and t denote the position of the site and time, respectively. P denotes the corresponding hopping probability of the pedestrians' move to the adjacent cell.

The dynamics of the Pedestrians are as follows. The Pedestrians enter and move through the crowded area towards the exit and Pedestrians do not overtake the Pedestrian in front of it. The entrance and exit are contralaterally positioned in the same row in the direction of abscissa. In this situation, it can be considered that the spatial dependencies of the Obstacles' position are eliminated. Thus, the dynamics of the Pedestrian and Obstacle can be extracted in a one-dimensional lattice since the Pedestrians move along a straight line between the entrance and exit. Fig. 1b illustrates the reduced one-dimensional model of the Pedestrians passing through a crowded area. The dynamics fundamentally follows the ASEP but interfered by obstacles. This kind of one-dimensional CA often used for modelling motor proteins [2] in which the Obstacles in this paper has a similar function as Attachment and Detachment.

The update rules for dynamical evolution of the system with corresponding hopping probabilities P are shown in Fig. 1c. By formulating these rules and considering the mean-field approximation and taking the continuous limit, the Burgers' equation can be derived. Comparison between the Burgers' equation derived from the ASEP and this model indicates that the Obstacle has an effect of reducing

the given hopping probability P . *i.e.* When the hopping probability of the ASEP is defined as P_{ASEP} , the relationship between the hopping probabilities of this model and obstacle density is derived as $P_{ASEP} = P (1 - \rho)$. The update algorithm of the pedestrian dynamics is as follows. Defining ρ as the density of the Obstacles in the crowded area shown in Fig. 1a. The density can be written as $\rho = N / ML$ when the number of Obstacle is defined as N ; and the obstacle density ρ means the probability of existence of an Obstacle in an arbitrary cell. Then the possible number of Obstacles in the one-dimensional lattice O is given as $O = N / M$. Subsequent to O obstacles being randomly allocated to empty cells at a given time, the lattice are randomly updated with the rules defined in Fig. 1c.

3 Simulations and Results

In order to investigate the effect of the crowd to the Pedestrian flow, entrance interval and the exit interval of the pedestrians are investigated. At first, it is assumed that Pedestrians periodically enter into the crowded area. This means that the intervals between each Pedestrian entering the system are constant. Possibilities exist that a Pedestrian may not be able to enter the entrance in the event of a congestion. Such a deadlock condition may occur if the density of Obstacle increases. Pedestrians “queue” rather than selecting to “call loss” at the entrance in order to avoid a deadlock, conserving the number of Pedestrians. This means that if a Pedestrian could not enter the entrance, the Pedestrian must wait outside until there is a space to enter. The simulation conditions are shown schematically in Fig. 2.

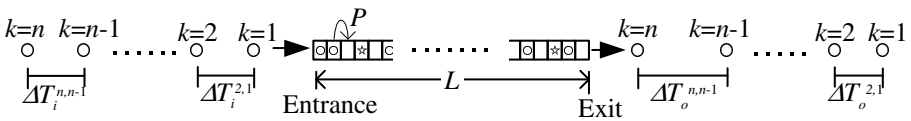


Fig. 2. The schematic interpretation of the simulation. Where k denotes the order of the Pedestrians entering and exiting the crowded area. $\Delta T_i^{n,n-1}$ and $\Delta T_o^{n,n-1}$ denote entrance interval and exit interval, respectively. Interval means that a time interval between $n-1$ th and n th Pedestrian.

If there is no Obstacle in the system, then the Pedestrian’s dynamics follows the ASEP and thus there would be no difference between the Pedestrian’s interval at the entrance and exit. In this study, two kinds of simulations are carried out to compare the exit interval of Pedestrians. Initially, the entrance interval $\Delta T_i^{n,n-1}$ is set at 30 steps with Obstacle density ranging from 0.01 to 0.4. Secondly, the entrance interval $\Delta T_i^{n,n-1}$ is set at 10 steps and the Obstacles density ranges from 0.1 to 0.4. For both cases the calculation is carried out for 50000 pedestrians and the hopping probability P of the Pedestrian is taken as $P = 1$ and the lattice size L is taken as 50. Fig. 3 illustrates the time-space diagrams obtained by this simulation of first 200 time steps for various Obstacles densities.

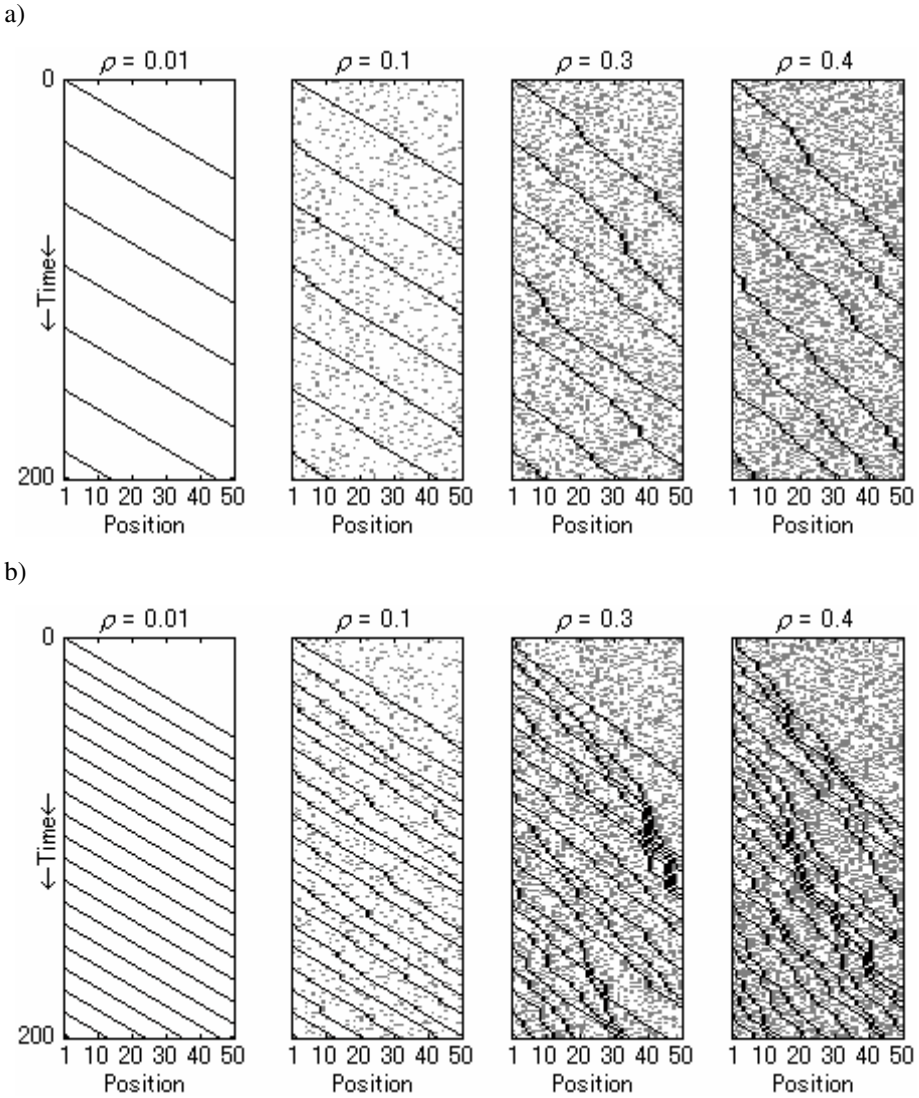


Fig. 3. Time-space diagram obtained by numerical simulations of first 200 time steps for various Obstacles density of $\rho=0.01$, $\rho=0.1$, $\rho=0.3$ and $\rho=0.4$. a) Entrance interval $\Delta T_i^{n,n-1} = 30$. b) Entrance interval $\Delta T_i^{n,n-1} = 10$. Where the dark solid line shows the track of Pedestrians and the grey speckles shows the Obstacles.

It is seen that the diagram shows the situation where Pedestrians walk through a crowded area. From the point of view of Pedestrian congestion, the travel time increases as Obstacles density increases due to the flow of Pedestrian interruption by the Obstacles. Further, shorter Pedestrian entrance interval causes larger interaction and congestion starts to occur.

Fig. 4 illustrates the resulting histogram of exit interval $\Delta T_o^{n,n-1}$ for each entrance interval case. Fig. 4a shows that the distribution of exit interval has a mean peak at 30 steps, which has the same value as the entrance interval and appears as a Gaussian type distribution. The variance of the distribution gets larger as Obstacle density increases. Although the Pedestrian flow is interrupted by Obstacles, there is no pedestrian traffic congestion since entrance interval is relatively large. Contrary to this, Fig. 4b illustrates the result where the entrance interval is set at 10 steps. The figure shows that when the Obstacle density is small, the histogram of the exit interval exhibits the same feature as the former case. As Obstacle density increases, the mean peak shifts to a shorter interval and a larger deviation is evidenced. This characteristic indicates that more Pedestrians exit within shorter intervals than the entrance intervals. The longer intervals indicate that some Pedestrians are cramped inside the crowded area for a longer time. These effects are resulting from the congestion of Pedestrians created by the Obstacles. This is because as the entrance interval becomes shorter, more friction is applied by the Obstacles. Note that deadlock at the entrance has not occurred for both 10 and 30 step interval simulations.

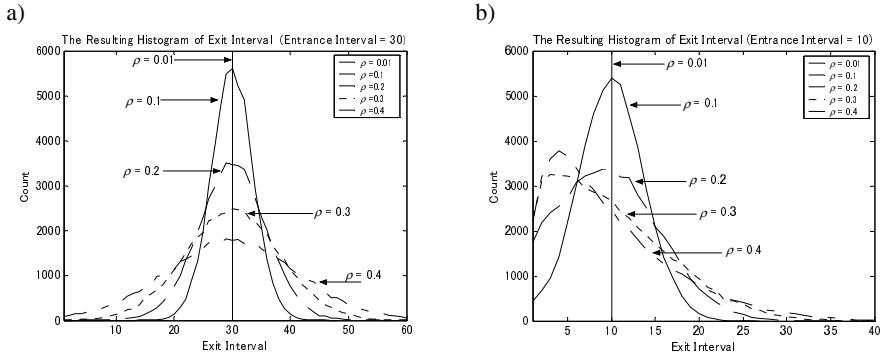


Fig. 4. The resulting histogram of exit interval for various obstacle density ρ . a) Entrance interval $\Delta T_i^{n,n-1} = 30$. b) Entrance interval $\Delta T_i^{n,n-1} = 10$.

4 Analytical Expression for Continuum Limit

The one-dimensional model described in the previous section is further investigated analytically to show how this ASEP-like model is relating with known ASEP. Fig. 5 illustrates the transition rules of the Pedestrian. The rules show the occurrence rules of the site i is occupied by a Pedestrian at time $t+1$. The notations used in the following sections are as follows. P denotes the hopping probability of a Pedestrian hops to the adjacent empty site. n_i^t denotes the existence probability of a Pedestrian at time t in site i . m_i^t denotes the existence probability of a Obstacle at time t in site i . E_i^t denotes the probability of the site i is empty. ρ denotes the density of Obstacles.

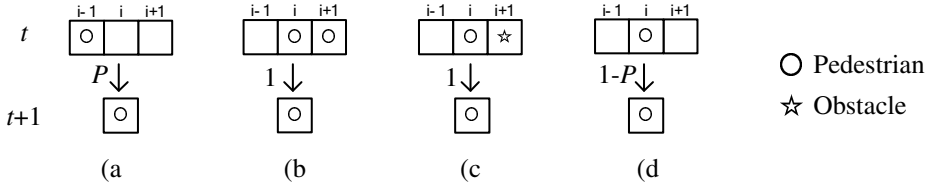


Fig. 5. The transition rules with corresponding transition probabilities. Where P denotes the hopping probability of pedestrian hopes to the adjacent empty site.

Then the normalising condition is defines as follows.

$$n_i^t + m_i^t + E_i^t = 1. \quad (1)$$

And m_i^t is expressed in terms of the Obstacle density ρ .

$$m_i^t = (1 - n_i^t)\rho. \quad (2)$$

The expression of existence probability of a Pedestrian at time $t+1$ in the site i (denoted as $n_i^{t+1}|_{\{a,b,c,d\}}$) for each rules shown in Fig. 5a, 5b, 5c and 5d are given as follows.

$$\begin{aligned} n_i^{t+1}|_a &= pn_{i-1}^t E_i^t. \\ n_i^{t+1}|_b &= n_i^t n_{i+1}^t. \\ n_i^{t+1}|_c &= n_i^t m_{i+1}^t. \\ n_i^{t+1}|_d &= (1 - p)n_i^t E_{i+1}^t. \end{aligned} \quad (3)$$

Taking the summation of Eq. 3 gives,

$$\begin{aligned} n_i^{t+1} &= n_i^{t+1}|_a + n_i^{t+1}|_b + n_i^{t+1}|_c + n_i^{t+1}|_d \\ &= Pn_{i-1}^t E_i^t + n_i^t n_{i+1}^t + n_i^t m_{i+1}^t + (1 - P)n_i^t E_{i+1}^t, \end{aligned} \quad (4)$$

Then using the expression of Eq. 1 and Eq. 2,

$$\begin{aligned} n_i^{t+1} &= Pn_{i-1}^t (1 - n_i^t - (1 - n_i^t)\rho) + n_i^t n_{i+1}^t + n_i^t (1 - n_{i+1}^t)\rho \\ &\quad + (1 - P)n_i^t (1 - n_{i+1}^t - (1 - n_{i+1}^t)\rho) \\ &= n_i^t + (P - \rho P)(n_{i-1}^t - n_i^t n_{i-1}^t - n_i^t + n_i^t n_{i+1}^t), \end{aligned} \quad (5)$$

Taking the ensemble average of Eq. 5 gives,

$$\langle n_i^{t+1} \rangle = \langle n_i^t \rangle + (P - \rho P) \langle (n_{i-1}^t - n_i^t n_{i-1}^t - n_i^t + n_i^t n_{i+1}^t) \rangle. \quad (6)$$

Assuming that there is no correlation between n_i^t and its neighbourhood.

$$\langle n_i^{t+1} \rangle = \langle n_i^t \rangle + (P - \rho P) \left(\langle n_{i-1}^t \rangle - \langle n_i^t \rangle \langle n_{i-1}^t \rangle - \langle n_i^t \rangle + \langle n_i^t \rangle \langle n_{i+1}^t \rangle \right). \quad (7)$$

Let $\langle n_i^t \rangle = u_i^t$ and rewriting Eq. 7 gives.

$$u_i^{t+1} - u_i^t = -(P - P\rho)(u_i^t - u_{i-1}^t) + (P - P\rho)u_i^t(u_{i+1}^t - u_{i-1}^t) . \tag{8}$$

The continuum limit of Eq. 8 can be derived by rewriting it into difference equation.

$$\begin{aligned} u(t + \Delta t, x) - u(t, x) &= -(P - P\rho)(u(t, x) - u(t, x - \Delta x)) \\ &\quad + (P - P\rho)u(t, x)(u(t, x + \Delta x) - u(t, x - \Delta x)) . \\ \text{where; } n_i^{t+1} &= u(t + \Delta t, x) \quad , \quad n_{i\pm 1}^t = u(t, x \pm \Delta x) . \end{aligned} \tag{9}$$

Then taking the limit of Δt and Δx to 0, Eq. 9 becomes as follows.

$$\frac{\partial u}{\partial t} \Delta t = (P - P\rho) \frac{\partial u}{\partial x} \Delta x - \frac{(P - P\rho)}{2} \frac{\partial^2 u}{\partial x^2} \Delta x^2 + 2(P - P\rho)u \frac{\partial u}{\partial x} \Delta x + O(\Delta x^3) . \tag{10}$$

Finally an expression in partial differential equation is derived as follows.

$$\begin{aligned} \frac{\partial u}{\partial t} &= (P - P\rho)C \frac{\partial u}{\partial x} - (P - P\rho)D \frac{\partial^2 u}{\partial x^2} + 2(P - P\rho)Cu \frac{\partial u}{\partial x} . \\ \text{where; } \Delta x / \Delta t &= C \quad , \quad \Delta x^2 / 2\Delta t = D . \end{aligned} \tag{11}$$

The analytically obtained expression of this CA model in continuum limit is shown in Eq. 11 which is known as the Burgers' equation. It is also known that the continuum limit of ASEP is also expressed in Burgers' equation as shown in Eq. 12 where P_{ASEP} denoted as the hopping probability of ASEP.

$$\frac{\partial u}{\partial t} = P_{ASEP}C \frac{\partial u}{\partial x} - P_{ASEP}D \frac{\partial^2 u}{\partial x^2} + 2P_{ASEP}Cu \frac{\partial u}{\partial x} . \tag{12}$$

Then the relationship between this model and ASEP can be seen by comparing the factors in each differential terms. Then the hopping probability P of this model can be transformed into P_{ASEP} by the following equation.

$$P_{ASEP} = P(1 - \rho) . \tag{13}$$

This expression indicates that this CA model can be transformed into the equivalent ASEP by using Eq. 13. Moreover, this CA model can also be interpreted as a model expressing the ASEP system with additional white noise. Also the consistency can be seen when there is no Obstacles, such that, $\rho = 0$ and $P_{ASEP} = P$ therefore Eq. 13 becomes Eq. 12. This result suggests that the relationship between this model and ASEP in a continuum limit is also applicable in the discrete system of CA.

Further, numerical investigation is performed to compare the behaviour of this CA model and the equivalent ASEP. Fig. 6 illustrates the time-space diagram to compare the evolution of this model and the equivalent ASEP. In this model, when $P = 1$, only the Obstacle has the effect to restrict the pedestrian's motion. Contrary to this, the hopping probability P has a significant meaning for restricting its motion in ASEP.

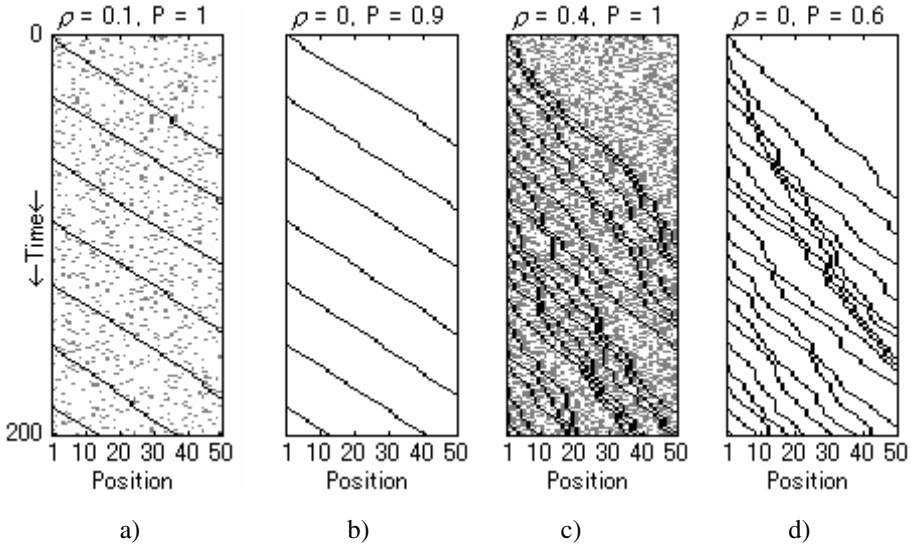


Fig. 6. Time-space diagram obtained by numerical simulations of first 200 time steps. a) Obstacle density $\rho=0.1$, $P=1$ and $\Delta T_i^{n,n-1} = 30$. b) Equivalent ASEP with $P = 0.9$ and $\Delta T_i^{n,n-1} = 30$. c) $\rho=0.4$, $P=1$ and $\Delta T_i^{n,n-1} = 10$. d) Equivalent ASEP with $P = 0.6$ and $\Delta T_i^{n,n-1} = 10$.

Fig. 7 illustrates the comparison of resulting histogram of exit intervals. The densities of Obstacle ρ are taken as 0.1 and 0.4 and hopping probability $P = 1$, whereas $P = 0.9$ and 0.6 in the equivalent ASEP. Again, the entrance intervals are taken as 30 and 10 then the exit intervals are counted. The solid line illustrates the result when Obstacle density is non-zero and the dashed line shows the result of corresponding ASEP where Obstacle density is zero. It is seen from the figure that the qualitative agreement between these two behaviours are fairly satisfactory.

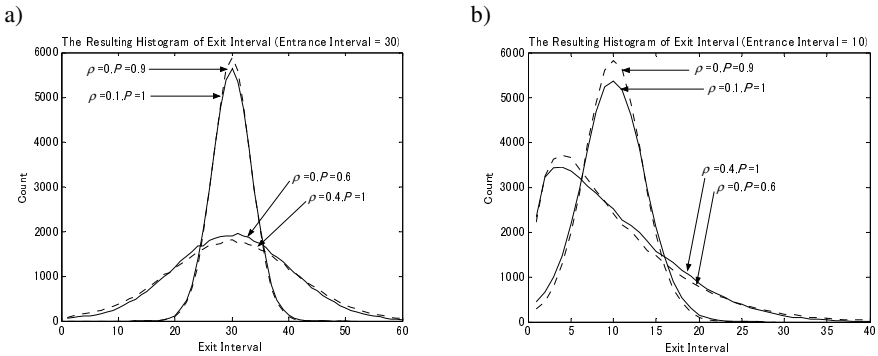


Fig. 7. The comparison of histograms of exit intervals obtained by this model and equivalent ASEP. a) Entrance interval $\Delta T_i^{n,n-1} = 30$. b) Entrance interval $\Delta T_i^{n,n-1} = 10$.

5 Averaged Pedestrian Flow in Periodic Boundary Condition

In addition, averaged Pedestrian flow under the periodic boundary condition is studied. When Obstacle density $\rho = 0$, flow Q is equivalent to that of ASEP. Let ϕ be the density of Pedestrian then Q is well given as shown in Eq. 14. [3]. Which is the expression of the Pedestrian flow for ASEP as well as this model when $\rho = 0$.

$$Q = \frac{1}{2} \left[1 - \sqrt{1 - 4P\phi(1 - \phi)} \right]. \tag{14}$$

In the previous section, the continuum limit for the open boundary condition is discussed and proved that this model is equivalent to the Burgers' equation. Knowing that the ASEP is also equivalent to the Burgers' equation in continuum limit, the hopping probability of ASEP can be expressed in terms of hopping probability and Obstacle density as shown in Eq. 13. Such relationship of the hopping rate between this model and ASEP is applied to obtain the averaged flow of this system. In Eq. 14, the term $(1 - \phi)$ shows the density of the empty site in ASEP. This aspect can be applied to derive how this term would be affected. The density of empty site of this model is expressed as $(1 - \phi - \rho)$ which has the same interpretation as the normalising condition shown in Eq. 1. Applying this expression to Eq. 14 gives as follows.

$$Q = \frac{1}{2} \left[1 - \sqrt{1 - 4P\phi(1 - \phi - \rho)} \right]. \tag{15}$$

Which is the expression of averaged flow of Pedestrians in this model. Further, the analytical expression of Eq. 15 is compared with numerical solutions by means of fundamental diagram. Fig. 8 illustrates the fundamental diagram of this model calculated both analytically and numerically under the periodical boundary condition. In the numerical simulation, the flow Q can be defined as the number of pedestrians moved in each time step divided by the total number of cells. In this simulation the total cells L are taken as 50 and Pedestrians are randomly allocated with given density at time $t=0$. Then count the number of Pedestrians moved at every update to obtain the flow at each time step. The averaged flow Q is obtained after 5000 time steps. The figure shows that the fundamental diagram for P is 1 and 0.6. Good agreement between the analytical and numerical solution is seen in each case.

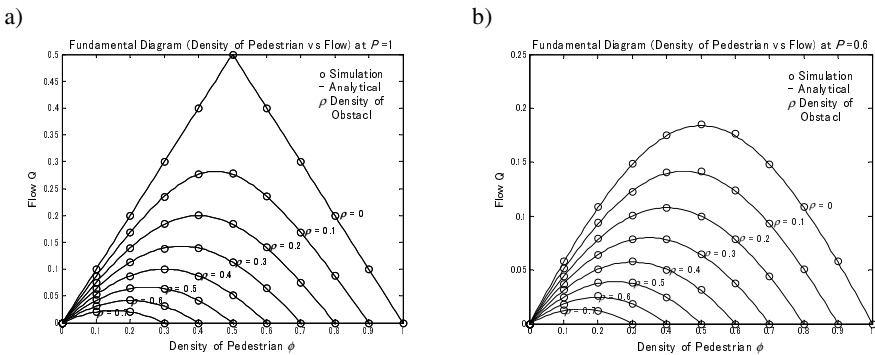


Fig. 8. The fundamental diagrams of this model for the case of a) $P = 1$ and b) $P = 0.6$. Where the solid line shows the analytical solution of Eq. 14 and the open circle shows the numerical results. The Obstacle densities of 0 to 0.7 are parametrically taken for each case.

6 Conclusions

In this study, pedestrian's motion in a crowded area is modelled in one-dimensional CA. When pedestrians enter a crowded area with a constant interval, the exit interval is highly affected by obstacle density. Variance on exit interval remains fairly small at low obstacle density. Whereas, congestion occurs and correlation among pedestrians increases when obstacle density is high. And also, analytically proven that this model can be expressed in Burgers' equation in the continuum limit and the relationship with well-known ASEP are derived. Moreover, the analytical expression of average flow of Pedestrian in periodic boundary condition that has a good agreement with numerical solution is shown. We believe that this model is an initiative and a starting point in improving city life and reducing congestion.

References

1. Nagatani, T.: The physics of traffic jams. *Rep. Prog. Phys.* 65, 1331–1386 (2002)
2. Nishinari, K., Okada, Y., Schadschneider, A., Chowdhury, D.: Intracellular Transport of Single-Headed Molecular Motors KIF1A. *PRL* 95, 118101-1–118101-4 (2005)
3. Schadschneider, A., Schreckenberg, M.: Cellular automaton models and traffic flow. *J. Phys. A: Math Gen.* 26, L679 (1993)

Excluded Volume Effect in a Pedestrian Queue

Daichi Yanagisawa^{1,2}, Yuki Tanaka¹, Rui Jiang³, Akiyasu Tomoeda⁴,
Kazumichi Ohtsuka⁵, Yushi Suma¹, and Katsuhiro Nishinari^{5,6,1}

¹ School of Engineering, The University of Tokyo, Japan

² JSPS Research Fellow, Japan

tt087068@mail.ecc.u-tokyo.ac.jp

http://www7b.biglobe.ne.jp/~daichi_yanagisawa/

³ School of Engineering Science, Univ. of Science and Technology of China, China

⁴ Meiji Institute for Advanced Study of Mathematical Sciences, Meiji Univ., Japan

⁵ Research Center for Advanced Science and Technology, Univ. of Tokyo, Japan

⁶ PRESTO, Japan Science and Technology Agency, Japan

Abstract. We have introduced excluded volume effect, which is a significant factor to model a realistic pedestrian queue, into queueing theory. The model has been exactly solved. Concretely, probability distributions and means of the number of waiting pedestrians, length of a queue, and waiting time have been derived. Due to the excluded volume effect, the process of closing up is included in our new model, so that the mean number of pedestrians increases as pedestrian arrival probability (λ) and leaving probability (μ) increase even if the ratio between them (i.e., $\rho = \lambda/\mu$) remains constant. Moreover, interval distance between pedestrians is included in our model because of the excluded volume effect, thus, length of a queue is considered more realistically than previous model. A queueing experiment is also performed to verify the validity of our model.

1 Introduction

Queue is an important phenomenon as evacuation in pedestrian dynamics [1] since it is observed everywhere in large cities and might cause congestion. Queueing theory [2], which has been considerably studied since Erlang started designing telephone exchanging system in 1909, is widely used to study many social systems [3]; however, it is not fully appropriate to apply for a pedestrian queue. In the *normal queueing model* (N-Queue) in the queueing theory, the state of a queue is represented by the number of waiting pedestrians and difference in interval distances between pedestrians are neglected. This identification makes it impossible to calculate correct length of a queue. Furthermore, in N-Queue, when there are some pedestrians in the queue, one pedestrian is always receiving service, and when he/she leaves the queue, the service for the next pedestrian starts immediately as shown in Fig. 1 (a). This phenomenon is suitable for a queue of packets in networks since operation for next packet starts instantly by a computer. However, it is not realistic for a queue of pedestrians since there is a delay of moving to service window as shown in Fig. 1 (b). Thus, we construct

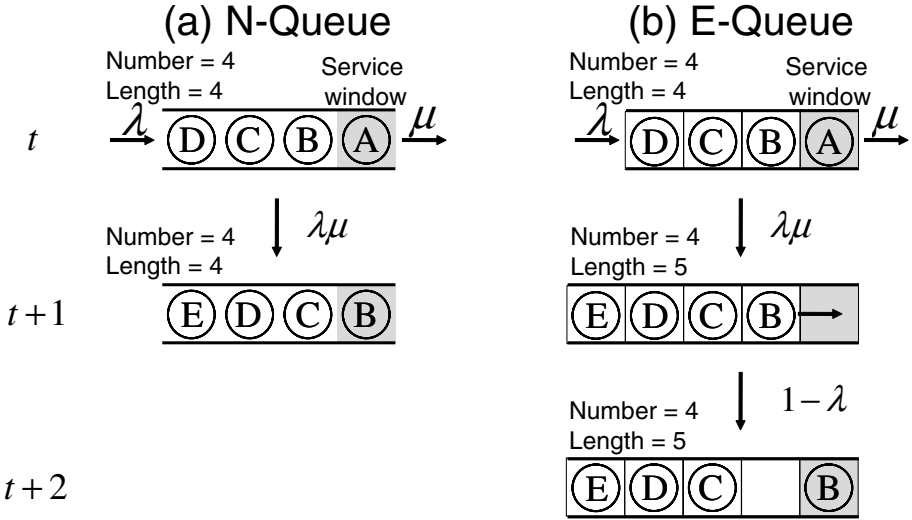


Fig. 1. Schematic views of N-Queue and E-Queue

the exclusive queueing model (E-Queue), taking the excluded volume effect into account, and compare it with N-Queue. We have also performed queueing experiment with real pedestrians to verify the validity of E-Queue model.

2 Queueing Models and Theoretical Analysis

2.1 Normal Queueing Model: N-Queue

We consider parallel dynamics models with discrete time in this paper because it is realistic for one dimensional pedestrian dynamics [4]. In N-Queue a pedestrian arrives at the queue with probability $\lambda \in [0, 1]$ and leaves with probability $\mu \in [0, 1]$ at each time step as shown in Fig. 1(a). The master equations of $P_N(n)$, which represents the probability that there are $n \in \mathbb{Z}_{\geq 0}$ pedestrians in the queue in the stationary state, could be obtained as follows:

$$P_N(0) = (1 - \lambda)P_N(0) + (1 - \lambda)\mu P_N(1), \tag{1}$$

$$P_N(1) = \lambda P_N(0) + (1 - \lambda)\mu P_N(2) + \{\lambda\mu + (1 - \lambda)(1 - \mu)\}P_N(1), \tag{2}$$

$$P_N(n) = \lambda(1 - \mu)P_N(n - 1) + (1 - \lambda)\mu P_N(n + 1) + \{\lambda\mu + (1 - \lambda)(1 - \mu)\}P_N(n) \quad (n \geq 2). \tag{3}$$

Note that the stationary state exists only when $\lambda < \lambda_{cr}(= \mu)$ is satisfied. λ_{cr} is a critical value of λ , and when $\lambda \geq \lambda_{cr}$, queue length tends to infinity. By solving these recurrence equations among three terms, we obtain $P_N(n)$ and $P_W(t)$ ($t \in [0, \infty)$) (Probability distribution of the waiting time, i.e., time between

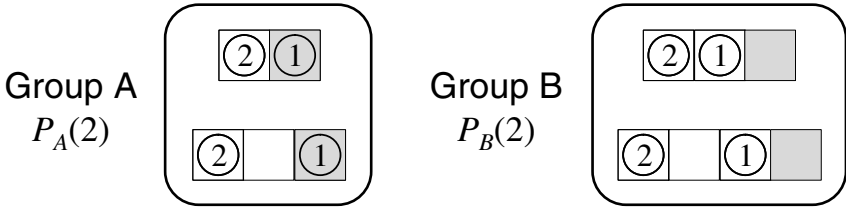


Fig. 2. Schematic views of the stationary states of E-Queue in the case $n = 2$. The service window is occupied by a pedestrian in the Group A, while it is vacant in the Group B.

a pedestrian arrives at the queue and leaves there). Then, mean number of pedestrians in the queue (N) and mean waiting time (W) are calculated as follows:

$$N = \sum_{n=0}^{\infty} nP(n) = \frac{(1 - \lambda)\rho}{1 - \rho}, \tag{4}$$

$$W = \sum_{t=0}^{\infty} tP_W(t) = \frac{1 - \lambda}{\mu(1 - \rho)}, \tag{5}$$

where $\rho = \lambda/\mu$ is ratio between the arrival probability and the service probability.

2.2 Exclusive Queueing Model: E-Queue

In E-Queue, the state is determined not only by pedestrian number n . Fortunately, due to deterministic movement of pedestrians in the queue (i.e., pedestrians move one cell in one time step if their proceeding cell is vacant), two consecutive vacant cells never appear in the stationary state. As a result, there are 2^n states when there are n pedestrians in the queue since we only need to consider whether there is a vacant cell or not in front of each pedestrian. Schematic views of the stationary states in the case $n = 2$ are depicted in Fig. 2.

Here, we focus on obtaining probability distributions of number of waiting pedestrians and waiting time, so that the 2^n states do not need to be distinguished completely. The important point is that whether the service window is occupied or not. Thus, the 2^n states are divided into two groups A and B. The service window is occupied in group A and is vacant in group B. For instance, two states belonging to group A, and the other two states belonging to group B in the case $n = 2$ as shown in Fig. 2.

We describe the sum of the probabilities of the stationary states in group A as $P_A(n)$ and that in group B as $P_B(n)$ when there are n pedestrians in the queue. Thus, $P_N(n) = P_A(n) + P_B(n)$. Note that in the case $n = 0$, we have $P_A(0) = 0$

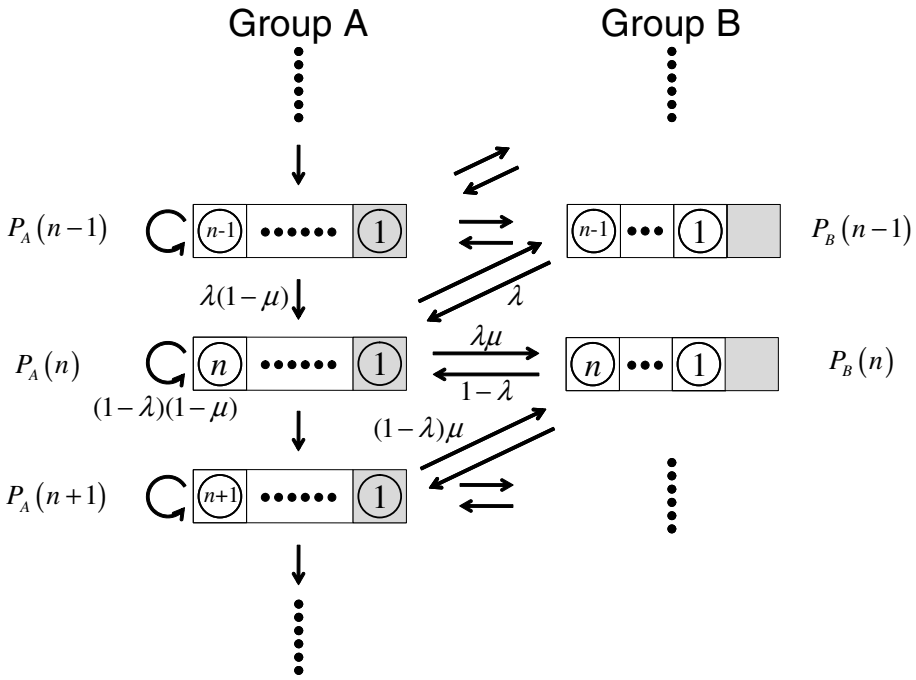


Fig. 3. State transition diagram of E-Queue

and $P_B(0) = P_N(0)$. The state transition diagram of E-Queue is depicted as Fig. 3 and the master equations in the stationary state are described as follows:

$$P_A(1) = (1 - \lambda)(1 - \mu)P_A(1) + \lambda P_B(0) + (1 - \lambda)P_B(1), \tag{6}$$

$$P_A(n) = \lambda(1 - \mu)P_A(n - 1) + (1 - \lambda)(1 - \mu)P_A(n) + \lambda P_B(n - 1) + (1 - \lambda)P_B(n) \quad (n \geq 2), \tag{7}$$

$$P_B(0) = (1 - \lambda)P_B(0) + (1 - \lambda)\mu P_A(1), \tag{8}$$

$$P_B(n) = \lambda\mu P_A(n) + (1 - \lambda)\mu P_A(n + 1) \quad (n \geq 1). \tag{9}$$

The probability distribution of the waiting time $P_W(t)$ is also calculated as

$$P_W(t) = f(t, 1)P(0) + \sum_{n=1}^{Q(t, 2)} [f(t - n, n)\mu P_A(n)] + \sum_{n=1}^{Q(t-1, 2)} [f(t - n, n + 1)\{(1 - \mu)P_A(n) + P_B(n)\}], \tag{10}$$

where $Q(a, b)$ returns a quotient of a/b , and $f(t, n) = \binom{t-1}{n-1} \mu^n (1-\mu)^{t-n}$ is the negative binomial distribution. Solving the equations (6-9) with normalization

condition $\sum_{n=0}^{\infty} P_N(n) = 1$ and using (10), we obtain the explicit form of $P_N(n)$ and $P_W(t)$ (see 5) in the case $\lambda < \lambda_{cr}(= \mu/(1 + \mu))$. The mean number of waiting pedestrians N and the mean waiting time W are obtained as

$$N = \frac{\rho}{1 - \frac{\rho}{1-\lambda}}, \tag{11}$$

$$W = \frac{1}{\mu \left(1 - \frac{\rho}{1-\lambda}\right)}. \tag{12}$$

The master equations where all 2^n states are distinguished are considered in 6 and the probability distribution of length of a queue, which includes the number of vacant cells, is obtained by solving them. The mean length of a queue L is described as

$$L = \frac{\rho}{1 - \lambda - \rho}. \tag{13}$$

3 Comparison between N-Queue and E-Queue

In this section, we compare physical quantities of N-Queue and E-Queue. The physical quantities are described by three parameters, which are λ , μ , and ρ . However, since $\rho = \lambda/\mu$, there are only two independent variables; thus, we use ρ and μ as independent ones in the following. Then, λ becomes a function of ρ and μ described as $\lambda(\rho, \mu) = \rho\mu$.

3.1 Critical Value λ_{cr}

In N-Queue, the critical value $\lambda_{cr} = \mu$. In E-Queue, $\lambda_{cr} = \mu/(1 + \mu)$. Since $\mu/(1 + \mu) \leq \mu$, the region where stationary state exists in $\rho - \mu$ space is smaller in E-Queue than in N-Queue. Due to time needed to close up vacant cells, which equals to one time step, the length of E-Queue diverges easier than that of N-Queue.

3.2 Mean Number N and Mean Length L

We compare N (mean number of pedestrians in the queue) in N-Queue and E-Queue, and L (mean length of the queue) in E-Queue in the stationary state. Figure 4 shows N and L against ρ . N and L monotonically increase with the increase of ρ . There are three remarkable points in Fig. 4. First, we can see that N in E-Queue is larger than that in N-Queue. In E-Queue there is delay of service since the waiting pedestrians have to proceed by one cell before they start to receive service, while there is no delay in N-Queue; therefore, the number of waiting pedestrians becomes larger in E-Queue than in N-Queue. Secondly, given ρ and μ , L is larger than N in E-Queue since L includes the length of vacant cells in the queue, whereas, N does not. Thirdly, we see that N in N-Queue decreases but N and L in E-Queue increase with the increase of μ . Since we adopt discrete

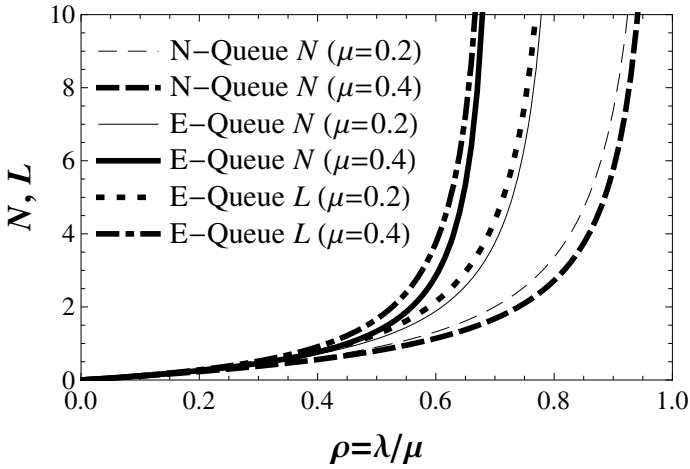


Fig. 4. Mean number N and mean length L against arrival-service ratio ρ

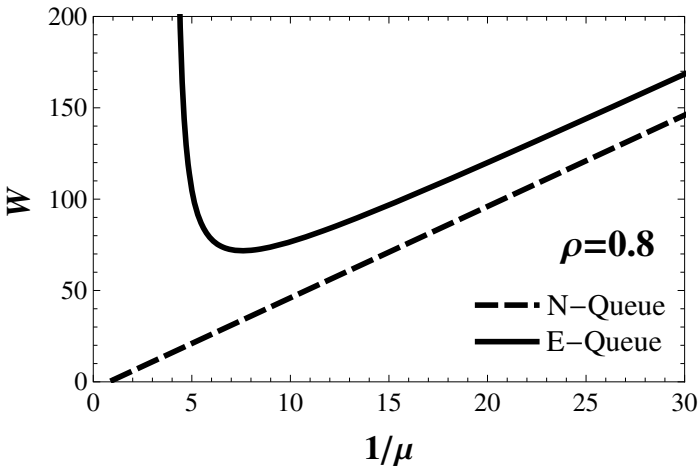


Fig. 5. Mean waiting time W against the mean service time $1/\mu$

time model, the variances of arrival and service time decrease with the increase of λ and μ . According to Pollaczek-Khintchine formula, Little’s theorem, and Ref. [2], N decreases as the variances of arrival and service time decrease, thus, N in N-Queue decreases when μ increases. In E-Queue, a pedestrian takes “the time of closing up” plus “the service time” to go through the service window. When μ increases the sum does not significantly decrease since the time of closing up remains as a constant. At given ρ , the increase of μ implies the increase of λ , hence, N and L increase as μ increases in E-Queue.

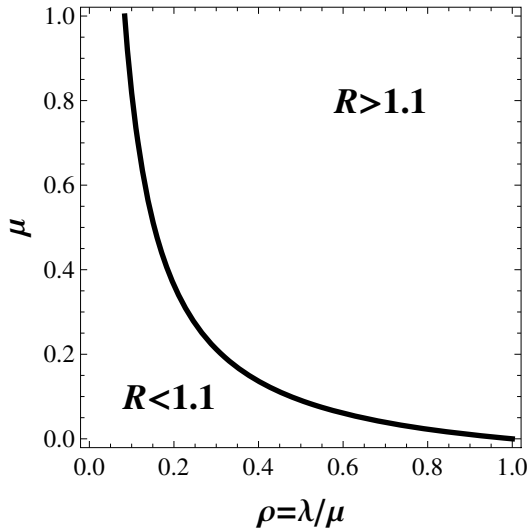


Fig. 6. Curve of $R = 1.1$ on the $\rho - \mu$ plane

3.3 Mean Waiting Time W

Figure 5 shows the variation of W against $1/\mu$, which is a mean service time, in the case ρ is constant. With the increase of $1/\mu$, W increases quasi-linearly in N-Queue, which coincides with our intuition. W also increases quasi-linearly in E-Queue when $1/\mu$ is large; however, when $1/\mu$ is small, it surprisingly achieves minimum $W_{\min}(= \rho/(1 - \sqrt{\rho})^2)$ at $1/\mu_{\min}(= \rho/(1 - \sqrt{\rho}))$, increases as $1/\mu$ further decreases, and diverges at $1/\mu_{\text{cr}}(= \rho/(1 - \rho))$. Since the Little’s theorem $N = \lambda W$ 2 is satisfied in all three models, $W = (1/\mu)(N/\rho)$. At a given ρ , with the increase of $1/\mu$, N increases in N-Queue, thus W increases. In contrast, N decreases with the increase of $1/\mu$ in E-Queue, hence, the minimum could be reached.

As we have seen in Fig. 5, E-Queue becomes similar to N-Queue when service time is large and different from it when service time is small. Thus, it is useful to know quantitatively when we should consider the excluded volume effect from the perspective of application. In Fig. 6, the $\rho - \mu$ plane is divided by the curve $R = 1.1$, where R is a ratio between W in N-Queue and E-Queue described as

$$R(\rho, \mu) = \frac{W_{\text{E-Queue}}}{W_{\text{N-Queue}}} = \frac{1 - \rho}{1 - (1 + \mu)\rho}. \tag{14}$$

In the lower-left region in Fig. 6 $R < 1.1$, and the difference of W is not critically large, so that it may be allowed to use N-Queue for simple calculation when both ρ and μ are small. In contrast, $R > 1.1$ and the difference is crucial in the upper-right region, therefore, the excluded volume effect should be considered when both ρ and μ are large. Note that Fig. 6 is an example of the dividing curve,

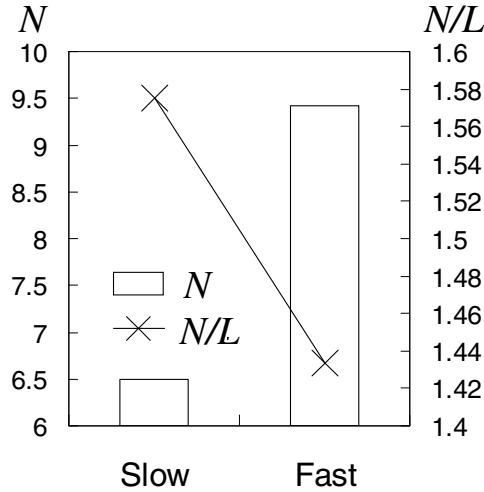


Fig. 7. Experimental mean number N and ratio between number and length N/L [m^{-1}]

and it is possible to depict the other curves by determining R as a different value. Thus, this diagram is helpful to know the error quantitatively and make a decision whether to use N-Queue or E-Queue for designing a queuing system for pedestrians.

4 Experiment

We have performed queuing experiments with real pedestrians to confirm that E-Queue model is more realistic. We have prepared the arrival gate and the service window. Computers give us random arrival time and service time. First, participants of the experiment stay at the arrival gate. When there is a signal from the computer, one of them gets in the queue. The participants proceed in the queue if there is enough space for them. Note that we do not tell them how to close up the queue. Thus, the queue is closed up in a natural way. Finally, they reach the service window and leave the queue when there is another signal from the computer. Two experimental conditions, which are *Slow* ($1/\lambda = 15$ [s], $1/\mu = 12$ [s], Small λ and μ) and *Fast* ($1/\lambda = 9.47$ [s], $1/\mu = 7.58$ [s], Large λ and μ) are performed. Note that $\rho = 0.8$ in both conditions.

We focus on the following two points: with the increase of μ (Slow \rightarrow Fast), (A) whether mean number of waiting pedestrians increases or decreases, (B) whether N/L remains constant or not. First, we consider (A). In Fig. 7, N obtained from videos of the experiment are shown. We surprisingly see that N is small in Slow case and large in Fast case. This implies that E-Queue is more realistic than N-Queue, since in E-Queue N increases with the increase of μ when

ρ is constant as we see in Sec. 3. Next, we examine (B). N/L is also depicted in Fig. 7. It is observed that N/L does not remain constant: it is large in Slow case and small in Fast case. N/L is constant in N-Queue; however, it is described as $1 - \lambda = 1 - \rho\mu$ in E-Queue and decreases as μ increases. Therefore, E-Queue is also more realistic than N-Queue from the point (B). N/L is considered as an average density in the queue. When μ is small (Slow case), pedestrians have enough time to close up the queue, so that the density is large. By contrast, when μ is large (Fast case), the service for a pedestrian finishes before all pedestrians in the queue close up, thus, the density becomes small. The two experimental phenomena are clearly reproduced in E-Queue since both the delay by closing up and the length of vacant cells, which are neglected in N-Queue, are considered. Therefore, E-Queue is more realistic model for pedestrian queueing system.

5 Conclusion

We have introduced the excluded volume effect into normal queueing model and exactly obtained the mean number of waiting pedestrians, length of the queue, and waiting time in the stationary state. Queueing experiment is also performed and its results agree well with the characteristics of our new queueing model with the excluded volume effect since it includes both the delay by closing up the queue and the interval distances between pedestrians.

Acknowledgment

We thank Kozo Keikaku Engineering Inc. in Japan for the assistance of the experiments. This work is financially supported by Japan Society for the Promotion of Science and Japan Science and Technology Agency. One of the authors (D. Y.) was also supported through the Global COE Program, “Global Center of Excellence for Mechanical Systems Innovation, by the Ministry of Education, Culture, Sports, Science and Technology.

References

1. Helbing, D.: Traffic and related self-driven many-particle systems. *Rev. Mod. Phys.* 73, 1067–1141 (2001)
2. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: *Queueing Networks and Markov Chains*. A Wiley-Interscience Publication, U.S.A (1998)
3. Helbing, D., Treiber, M., Kesting, A.: Understanding interarrival and interdeparture time statistics from interactions in queueing systems. *Physica A* 363, 62–72 (2006)
4. Rogsch, C., Schadschneider, A., Seyfried, A., Klingsch, W.: How to select the “right one” - update schemes for pedestrian movement in simulation and reality. In: *Proceedings of the Traffic and Granular Flow’09* (to be published)
5. Yanagisawa, D., Tomoeda, A., Jiang, R., Nishinari, K.: Excluded volume effect in queueing theory. *JSIAM Letters* (2010) (to be published), e-print arXiv:1001.4124
6. Yanagisawa, D., et al.: (in preparation)

Simulation on Vehicle Emission by the Brake-Light Cellular Automata Model

Liyun Dong, Peng Zhang, and Shiqiang Dai

Shanghai Institute of Applied Mathematics and Mechanics, Shanghai University,
Shanghai, 200072, P.R. China
dly@shu.edu.cn

Abstract. Vehicle emission has become a major source of air pollution. In this paper, the brake-light cellular automaton model incorporated with a vehicle emission model is utilized to investigate emitted exhaust pollution by traffic flow. First, both macro- and microscopic features of traffic flow are reproduced quantitatively and compared with empirical findings. Then the model is used to simulate the vehicle emission of a moving fleet of vehicles. It is shown qualitatively that the emission rate is significantly increased in the medium density range with considering instantaneous velocity and acceleration together. Usually the total amount of pollutant discharge from vehicles is underestimated by considering average velocity alone. It is believed that a good driving strategy, e.g. eco-driving, is an effective way to reduce vehicle emission.

Keywords: cellular automata model, vehicle emission, free flow, traffic jam.

1 Introduction

In recent years, urban air pollution is increasingly getting serious and constitutes a great threat to human health. In big cities, vehicle emission has become a major source of air pollution. Over 60% CO and over 40% NO_x come from vehicle sources. Detrimental air pollution generated by traffic flows on roads has reached the critical level in many cities, especially in downtown areas. The reduction of air pollution, mainly caused by the high level of vehicle emissions, is a prime requisite for the future sustainable development of a green transportation.

It is well known that the vehicle emission is closely related to the state of traffic flow. The transient driving patterns formed by the repeated decelerations and accelerations of vehicles often occur at intersections or in self-organized jamming phase on roads (e.g., the stop-and-go traffic), produce substantial additional amounts of emitted exhaust pollution. It is found that even for vehicle trips with the same average velocities, their fuel consumption and emissions rates are significantly different. A variable velocity trip emits much more exhausts than a constant velocity trip. Hence, it is necessary to investigate the detailed evolution of vehicular flow by microscopic models, which provides a powerful way to predict the vehicle emission. Micro-simulation of traffic flow is able to give detailed description of a queue of moving vehicles. Traditionally, car-following

models were used to simulate the behaviors of single vehicles. Recently, cellular automaton (CA) models begin to serve as a promising method for micro-simulation, which are discrete in both space and time. The Nagel-Schreckenberg model (cited as “the NaSch model” for short) is a probabilistic CA model for the description of single-lane highway traffic [1]. Although the model is simple and involves fewer parameters, it is able to reproduce the basic phases in real traffic, i.e., free flow and traffic jam. Since then this model has been generalized in many aspects and applied widely in studying various traffic phenomena in reality [2]. Compared with other models, the CA models are simple, flexible and suitable for parallel computation of traffic flow.

However, as was pointed out by Knosp et al [3], most of the existing models, fail to reproduce the microscopic features observed in real traffic. The comparison of simulation results with empirical data at a microscopic level is not so satisfactory. Recently, new CA models have been proposed to depict the synchronized flows, which is another basic traffic phase in real traffic [4]. The brake-light CA model proposed by Knosp et al [5] was based on the NaSch model but introduced the effect of brake light and velocity anticipation. Therefore, this model can reproduce all three traffic phases and provide better agreement with empirical findings [6]. However, to the authors' knowledge, most of CA models have not been used to evaluate the emission from vehicle fleets. Therefore, it is worthy to propose a dynamic emission model, which combine vehicle emission model with a proper CA model.

In this paper, we investigate the exhaust pollution emitted from a vehicle fleet with the brake-light CA model incorporated with a vehicle emission model. We discuss some macro- and microscopic features given by simulation results with the CA model. More attention will be paid on microscopic features of traffic flow, especially the variation of acceleration, which is closely related to the vehicle emission. Then we calculate the average emission rate of two typical pollutants by taking instantaneous velocity and acceleration into account. The simulation results are compared with those merely considering average velocity to show the effectiveness of the presented model.

2 Traffic Flow Model with Emission

The dynamic emission model for vehicle fleet consists of two main parts: a microscopic CA model and a microscopic emission model. The brake-light CA model (the BL model for short) is adopted in the paper, which is defined on a one-dimensional lattice of L cells with periodic boundary condition. The length δ_x of a cell is given by 1.5 m. Each vehicle has the same dynamical length $l = 5$ cells, i.e., 7.5 m in average. Vehicles move from the left to the right on a lane, and they are numbered in the driving direction. The state of the n -th vehicle is characterized by its position $x_n(t)$ and speed $v_n(t)$ ranging from 0 to v_{\max} at time t . The gap between consecutive cars is defined as $d_n(t) = x_{n+1}(t) - x_n(t) - l$ and b_n is the status of the brake light (on/off $\rightarrow b_n = 1/0$). At each discrete time-step, these vehicles are updated in parallel according to the following rules:

Step 0. Determining the randomization parameters

$$p = p(v_n(t), b_{n+1}(t)) = \begin{cases} p_b, & \text{if } b_{n+1} = 1 \text{ and } t_h < t_s \\ p_0, & \text{if } v_n = 0 \\ p_d, & \text{in all other cases} \end{cases}$$

$$b_n(t+1) = 0$$

The randomization parameter p is determined by the current velocity of the n -th car and the status of the brake light of the preceding car. The two times $t_h = d_n/v_n(t)$ and $t_s = \min(v_n(t), h)$ are introduced, where h determines the range of interaction with brake light. The braking parameters p_d , p_b and p_0 for cars are determined according to different states (See [4] for details).

Step 1. Driving as fast as possible:

$$\text{if } (b_{n+1}(t) = 0 \text{ and } b_n(t) = 0) \text{ or } (t_h \geq t_s) \rightarrow v_n(t') = \min(v_n(t) + 1, v_{\max})$$

Step 2. Braking due to safety consideration:

$$v_n(t'') = \min(d_n^{eff}, v_n(t'))$$

$$\text{if } (v_n(t'') < v_n(t)) \text{ then } b_n(t+1) = 1.$$

The effective gap $d_n^{(eff)}$ is defined as $d_n + \max(v_{ami} - gap_s, 0)$, where the expected velocity of the preceding car $v_{ami} = \min(d_{n+1}, v_{n+1})$ and the parameter gap_s is used to control the effectiveness of the anticipation.

Step 3. Including noise due to individual variations of driver behavior:

$$\text{if } (rand() < p) \text{ then } v_n(t+1) = \max(v_n(t'') - 1, 0)$$

$$\text{if } (p = p_b \text{ and } v_n(t+1) = v_n(t+1) - 1) \text{ then } b_n(t+1) = 1.$$

Step 4. Giving vehicle movement:

$$x_n(t+1) = x_n(t) + v_n(t+1)$$

Step 5. Determining acceleration:

$$a_n(t+1) = v_n(t+1) - v_n(t)$$

Step 6. Calculation of vehicle emission

$$E_n(t) = G(v_n(t+1), a_n(t+1))$$

where a_n and E_n are the acceleration and emission rate of the n th vehicle respectively.

G is a function of instantaneous velocity and acceleration, which measures the emission of different pollutants. In general, emission rates can be expressed as a function of the type and age, average velocity, operating mode and other properties of vehicles. Here we only consider the effect of velocity and acceleration on emission rates.

A general function for all pollutant emission is given as

$$E_n(t) = G(v_n(t), a_n(t))$$

$$= \max(E_0, f_1 + f_2 v_n(t) + f_3 v_n^2(t) + f_4 a_n(t) + f_5 a_n^2(t) + f_6 v_n(t) a_n(t)) \tag{1}$$

where f_1 to f_6 are emission constants specified for each vehicle and pollutant type determined by the regression analysis. Two main pollutants emitted from cars and

heavy duty vehicles (HDV, diesel) were modeled, i.e., carbon dioxide (CO₂) and volatile organic compounds (VOC). These particular pollutants were chosen based on their potential health impacts and external costs [7].

The functions (including the lower emission limit E_0 and the constants) are determined for the two pollutants and for cars and HDVs. These coefficients are listed in Table 1. Usually the lower emission limit E_0 is set to zero. Then each function can be used to predict the emissions of a car and a trip not included in the data set used for the regression analysis to check the accuracy of the predicted value [7].

Table 1. Coefficients of emission function for car and hdv

Pollutant	f_1	f_2	f_3	f_4	f_5	f_6
CO ₂ (CAR)	5.53e-1	1.61e-1	-2.89e-3	2.66e-1	5.11e-1	1.83e-1
VOC(CAR) a≥-0.5 m/s ²	4.47e-3	7.32e-7	-2.87e-8	-3.41e-6	4.94e-6	1.66e-6
VOC(CAR) A<-0.5 m/s ²	2.63e-2	0	0	0	0	0
CO ₂ (HDV)	1.52	1.88	-6.95e-2	4.71	5.88	2.09
VOC(HDV)	1.04e-3	4.87e-4	-1.49e-5	1.27e-3	2.10e-4	1.00e-4

3 Simulation and Discussion

The parameters of the BL model were calibrated by empirical data and their values are $v_{\max} = 108$ km/h, $p = 0.1$, $p_0 = 0.5$, $p_b = 0.94$, $gap_s = 7$ and $h = 6$ respectively. We simulate a system with $L = 10^4$ cells, which corresponds to the length of the actual lane about 15 km. The global density under the periodic condition is given by

$$\rho = 1000 \times N / (L \times \delta x) \quad (2)$$

The mean velocity

$$\bar{v} = \frac{1}{NT} \sum_{t=T_0}^{T+T_0-1} \sum_{n=1}^N v_n(t) \quad (3)$$

and the flux is calculated by

$$q = \rho \bar{v} \quad (4)$$

The fundamental diagram is obtained by this formula. Here N denotes the total number of cars in the system. Each run of simulation is first conducted $T_0 = 10^5$ time steps to reach the stationary state and the data are recorded in successive $T = 10^5$ time steps.

The mean emission rate is calculated by

$$\bar{E} = \frac{1}{NT} \sum_{t=T_0}^{T+T_0-1} \sum_{n=1}^N E_n(t) \quad (5)$$

And the emission rate calculated from average velocity \bar{v} is give by

$$E_{\bar{v}} = G(\bar{v}, 0) \quad (6)$$

Most of numerical results are obtained from the random initial condition, i.e., vehicles are randomly distributed on the lane and their speeds are set to be zero at the beginning

of simulations. And the system is investigated under the periodic condition, i.e., a fleet of vehicles run on a ring road.

3.1 Macro- and Microscopic Features

Firstly we reproduce some results to show the basic features of the BL model reported in [5] and [8]. Then we provide the velocity distribution of all cars in the steady states, which give a qualitative description of traffic phases in the spatial-temporal diagram. More attention is paid on the distribution of acceleration, which is a key to get a correct evaluation of emission for vehicle fleet.

The fundamental diagram for the BL model is given in Fig. 1, which is obtained from three different initial conditions, i.e., random (R), homogeneous (H) and mega-jam (M) distributions. It is shown that the BL model is not sensitive to different initial conditions as the NaSch model.

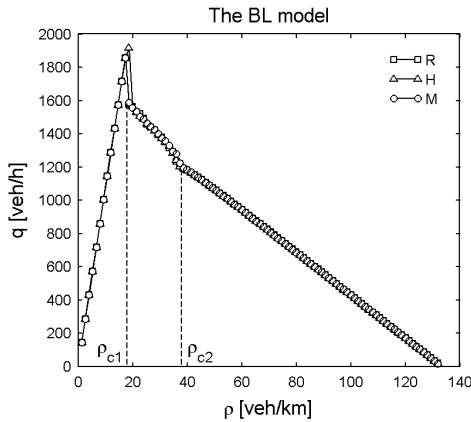


Fig. 1. Fundamental diagram via numerical simulation from a random initial distribution. The discontinuity is due to the finite size effect.

Three density regimes can be distinguished. (1) For the density lower than ρ_{c1} , it is the free flow. (2) For the density between ρ_{c1} and ρ_{c2} , it is the coexistence state of free flow and congested flow, the latter consisting of synchronized flow and traffic jam. (3) For the density higher than ρ_{c2} , the free flow disappears and only synchronized flow and traffic jams coexist.

The one-minute average data sampled locally by a virtual detector are plotted in Fig. 2. From the local fundamental diagram, the free flow branch is reproduced quite well. The slope is in agreement with the empirical findings [6]. For congested traffic, the model can reproduce some features of synchronized traffic. This interpretation of the flow data is supported by measurements of the cross-correlation function [5]. In the presence of wide jams the flow is proportional to the densities as was found by empirical observation. In real measurements the branch extends up to quite high densities 70 veh/km. The simulation result coincides with this observation, which is better than that in the NaSch model.

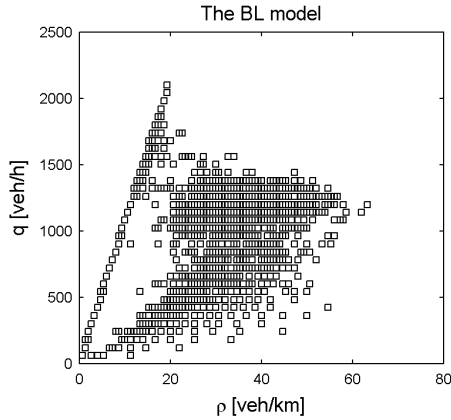


Fig. 2. Local fundamental diagram via numerical simulation from a random initial distribution

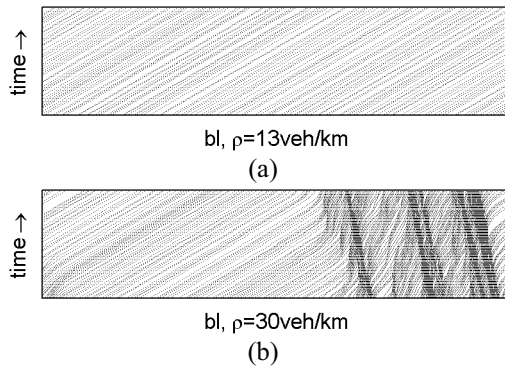


Fig. 3. The space-time plot in the case of different densities. (a) $\rho = 13$ veh/km, (b) $\rho = 30$ veh/km

From the space-time plot in Fig. 3, we can find the free flow qualitatively (Fig. 3a). In contrast to the NaSch model, it is clearly shown that there exists the coexistence of free flow, synchronized flow and jam in Fig. 3b.

The quantitative description of the steady state in the case of typical densities is given in the Fig.4, i.e., the distribution of velocity of all cars in the system. It is found that there are new peaks around the velocity 5 in the congested flow instead of only two peaks for free flow ($v = v_{\max}$) and jam ($v = 0$). It is a signal that there may exist a new phase and not merely a transient phase between free flow and jam. The new phase is actually identified as the synchronized flow according to more criteria. It is also confirmed that the BL model can reproduce heavy synchronized flow (with low velocities) and fails to reproduce light synchronized flow (with high velocity) [8].

Usually, more attention has been paid on the flux or average velocity of vehicles in simulation of traffic flow. However, the acceleration of vehicles plays a vital role in predicting the emission of vehicles. In fact, vehicle emissions are more sensitive to acceleration than to velocity.

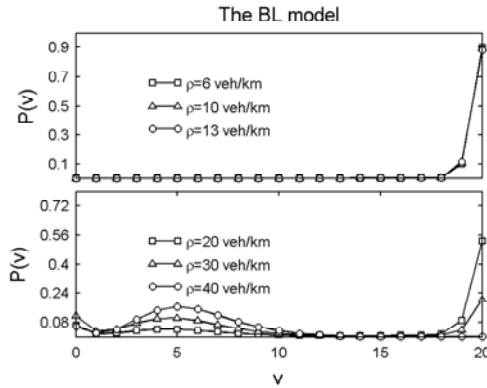


Fig. 4. Distribution of speed of all cars in the BL model in free flow and congested flow for different densities when the system reaches the steady state

The acceleration distributions of the BL model is shown in Fig. 5. For free flow traffic, the distribution is almost symmetrical due to random deceleration. For congested traffic, the distribution becomes asymmetrical and larger decelerations appear when a high-velocity vehicle is approaching its preceding vehicle at rest. The follow-the-leader data reveals that empirical accelerations and decelerations are usually limited to the range between -3 and $+4$ m/s^2 [9]. The simulation results are in agreement with empirical facts.

The fractions of acceleration at different densities are plotted in Fig. 6, where three regions were distinguished by the sign of acceleration. For lower densities ($\rho < \rho_{c1}$), the ratio of acceleration ($a > 0$) is almost equal to that of deceleration ($a < 0$) which is due to randomness and less interaction between successive cars. For medium densities $\rho_{c1} < \rho < \rho_{c2}$, both fractions of acceleration and deceleration states are increasing significantly due to the existence of three phases. For higher densities ($\rho > \rho_{c2}$), both of them begin to decrease and constant-velocity states increase due to the growth of jam.

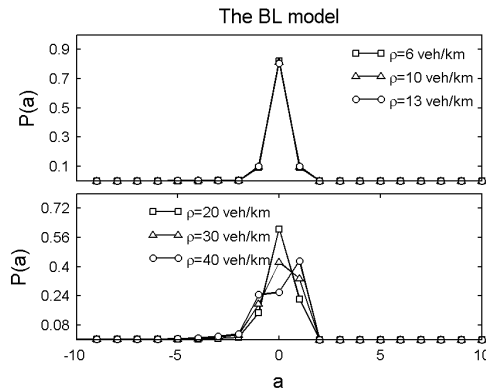


Fig. 5. Distribution of acceleration of all cars in the BL model in free flow and congested flow for different densities when the system reaches the steady state

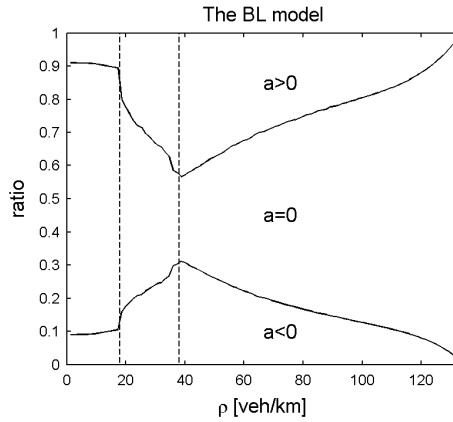


Fig. 6. Fraction of acceleration for all cars in the BL model for different densities when the system reaches the steady state

3.2 Vehicle Emission

Fig. 7 and 8 show the emissions of CO_2 and volatile organic compounds (VOC) for cars and heavy duty vehicles (HDVs), in which numerical results are calculated by both instantaneous velocity and acceleration. The results calculated by average velocity alone are shown together for comparison. As was expected, emissions from HDVs are significantly larger than that from cars. In general, the emissions for most pollutants, e.g., CO_2 , PM etc. calculated by instantaneous velocity and acceleration (Eq. 5) is larger than that calculated by average velocity (Eq. 6), since acceleration or deceleration usually leads to additional emissions. It is not the case that the emission of VOC for HDVs calculated by average velocity is higher for the density between ρ_{c1} and ρ_{c2} . The reason is that the states with low constant-velocity generate considerable emission of VOC and negligible emission of other pollutants.

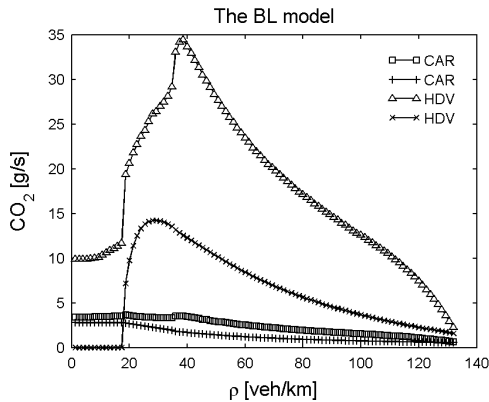


Fig. 7. Average emission rate of CO_2 as a function of the density. The squares and triangles indicate the numerical results obtained with Eq. (5) and the others with Eq. (6)

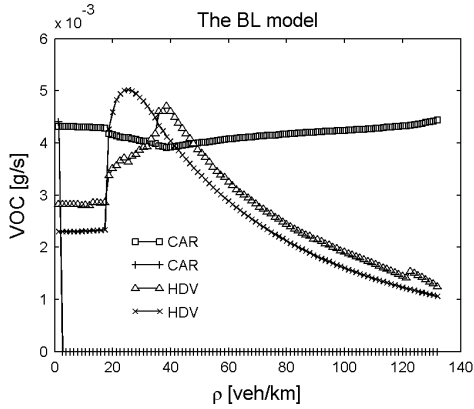


Fig. 8. Average emission rate of VOC as a function of the density. The squares and triangles indicate the numerical results obtained by Eq. (5) and the others by Eq. (6)

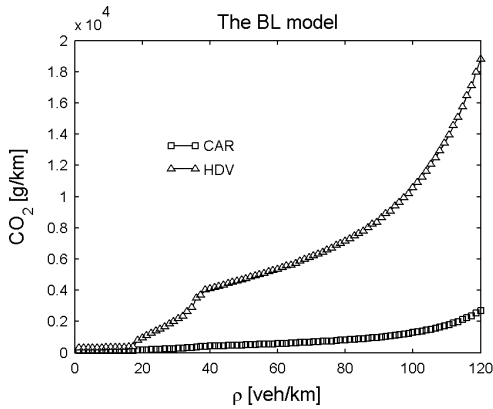


Fig. 9. Emission factor of VOC as a function of the density

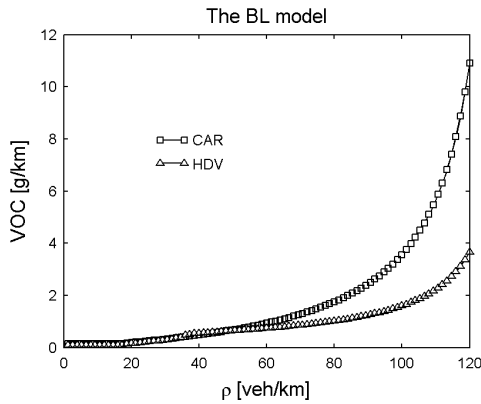


Fig. 10. Emission factor of PM as a function of density

It is convenient to convert the above results from g/s to g/km by calculating $1000\bar{E}/\bar{v}$. Figures 9 and 10 show the emission amount of CO₂ and VOC for different velocities in kilometers respectively, which are calculated by instantaneous velocity and acceleration. It is shown that the free flow states generate least emissions for lower densities. With increasing density, more and more emissions generate. It is consistent with daily experience. In contrast to most exhaust pollution generated by HDVs, they produce less emission of VOC than cars. This results can be confirmed by Fig. 8 that the emission rate of VOC from HDVs is more than that from cars in most cases.

4 Conclusions

We have simulated vehicle emission with the BL model, which has been calibrated with empirical facts. The BL model can exhibit all three phases of traffic and reproduce reliable numerical results compared with real traffic. Then we investigate some macro- and microscopic features simulated by this model. Furthermore, we evaluated the emission rate of two typical pollutants generated by cars and HDVs. Numerical simulations show that the model can capture the main features of emission from vehicles. It is turned out that emissions are more sensitive to the level of vehicle acceleration than to vehicle velocity. It is concluded that calculating the vehicle emission by average velocity alone usually underestimate the total amount of pollutant discharge. Experimental investigations should be performed to verify numerical results. Moreover, good driving styles (e.g., eco-driving) will lead to the reduction of vehicle emission. It should be studied further to reduce air pollution with better driving styles.

Acknowledgments. This work was supported by the 973 Program (Grant No. 2006CB705500), the NSFC projects (Grant No. 10672098 and No. 10972135).

References

1. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. *J. Phys. I(France)* 2, 2221–2233 (1992)
2. Chowdhury, D., Santen, L., Schadschneider, A.: Statistical physics of vehicular traffic and some related systems. *Phys. Rept.* 329, 199 (2000)
3. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: Empirical test for cellular automaton models of traffic flow. *Phys. Rev. E* 70, 016115 (2004)
4. Kerner, B.S.: Complexity of synchronized flow and related problems for basic assumptions of traffic flow theories. *Networks and Spatial Economics* 1, 35–76 (2001)
5. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: Towards a realistic microscopic description of highway traffic. *J. Phys. A* 33, L477–L485 (2000)
6. Neubert, L., Santen, L., Schadschneider, A., Schreckenberg, M.: Single-vehicle data of highway traffic: A statistical analysis. *Phys. Rev. E* 60, 6480 (1999)
7. Panis, L.I., Broekx, S., Liu, R.: Modelling instantaneous traffic emission and the influence of traffic speed limits. *Science of the Total Environment* 371, 270–285 (2006)
8. Jiang, R., Wu, Q.S.: Cellular automata models for synchronized traffic flow. *J. Phys. A* 36(2), 381–390 (2003)
9. Helbing, D., Tilch, B.: Generalized force model of traffic dynamics. *Phys. Rev. E* 58(1), 133–138 (1998)

Bidirectional Traffic on Microtubules

Maximilian Ebbinghaus^{1,2}, Cécile Appert-Rolland^{2,3}, and Ludger Santen¹

¹ Fachrichtung Theoretische Physik, Universität des Saarlandes,
D-66123 Saarbrücken, Germany
ebbinghaus@lusi.uni-sb.de,
l.santen@mx.uni-saarland.de

² Univ. Paris-Sud, Laboratoire de Physique Théorique, UMR8627,
Bâtiment 210, Orsay F-91405, France
cecile.appert-rolland@th.u-psud.fr

³ CNRS, Orsay F-91405, France

Abstract. Intracellular transport involves the processive displacement of molecular motors on microtubules. These motors are specialized to walk in one or the other direction on the microtubule. It is not known yet how this bi-directional traffic is organized in order to be efficient. Here we use some modeling based on cellular automata models to point out the problems caused by bidirectional transport, we discuss the role of confinement around the microtubule, and we illustrate how the dynamics of the microtubules could help preventing jam formation.

Keywords: intracellular transport, bidirectional transport, dynamical network, cellular automata simulations.

1 Introduction

Biological cells are complex objects, which are kept alive by a multitude of active processes. They are the elementary building blocks of complex organisms but also complex objects themselves with their own transport infrastructure.

The transport is driven by specialized proteins, which enable objects on the nano- and microscale e.g. to pass barriers and to be transported over large distances. The long distance transport is carried out by molecular motors which are using the cytoskeleton, i.e. the intracellular filament network, as tracks. Here we concentrate on transport on microtubules, which are of particular importance for the transport from the nucleus to the membrane.

Microtubules are cylindrical structures resulting from the polymerization of tubulin units. They are polarized, and as a result, some motors - such as kinesins - walk predominantly towards the plus end of the microtubule, while some others - e.g. dyneins - process towards the minus end. It is not understood yet how nature organizes such a bi-directional flow in order to get efficient transport. On the other hand, such a knowledge would be of great interest, first from a fundamental point of view to understand the mechanisms taking place in a living cell, but also because it has been observed that some jamming of molecular motors is occurring in axons in some neuronal diseases such as Alzheimer [\[8\]](#).

The processive motion of molecular motors is performed through cyclic changes of conformation of the motors. One cycle can be called a “step”. The typical amplitude of such a step is of the order of one tubulin unit, i.e. 8 nanometers, while the total length of the microtubule can range from a few to hundreds of microns. Due to the discrete nature of the steps, it has become now usual to model motor motion with cellular automata.

In cellular automata models, a microtubule is represented by a one dimensional lattice. Motors attach and detach with certain rates, and in the meantime hop from site to site with again appropriate probabilities. First models were monodirectional, a setup that is appropriate for motility assays ([4,6,5,9]). The intracellular traffic between cell center and membrane however is bidirectional.

Also bidirectional stochastic transport on one-dimensional lattices has been extensively investigated. Compared to the realistic intracellular motion of molecular motors, most models differ in some respect. Many models allow for particle exchanges on the track. Klumpp and Lipowsky [3] introduced a model for two species of motors processing in opposite direction. However, the diffusive reservoir around the microtubule was not represented explicitly: after detachment, motors can attach again anywhere in the system, i.e. there is no memory of their previous attachment point. This could be a good representation for an isolated microtubule in an infinite empty space. However, the cell is not empty at all. Many obstacles create steric constraints. Molecular motors are rather small, but the cargos that they carry are often much larger (e.g. vesicles or mitochondria), and thus may not diffuse very far from the microtubule.

In this paper, we shall present how a jamming transition is occurring when the diffusive reservoir confined around the microtubule is explicitly represented by a second lane in the model - at least beyond a certain density of motors. We shall discuss how this transition is modified when the confinement constraint is more or less relaxed. Then we shall show how, in the confined case, an efficient flow can be recovered when the microtubule is represented by a dynamical filament instead of a static one.

2 Definition of the Model

In this section, we consider only static filaments, as in [2]. The case of dynamical filaments will be considered only in section 5.

The microtubule is represented by a one-dimensional lattice of size L . As we are interested in the bulk transport properties, we consider periodic boundary conditions. On this lattice, motors undergo processive motion. Two types of motors are considered, moving in opposite directions. We shall refer to them as positive and negative motors. During an infinitesimally small time step dt , each motor can hop to the next site in its processive direction with probability $p dt$, if the target site is empty. An exclusion constraint imposes that no more than one motor can occupy a given site on the microtubule. As a consequence, motors moving in opposite directions cannot pass each other unless at least one of them leaves the microtubule.

The surroundings of the microtubule is represented by another one-dimensional lattice. On this second lane, there is no exclusion principle, and the motors undergo some symmetric diffusion motion: during a time step dt , they can jump to their left or to their right with equal probability $D dt$.

Motors go from one lane to the other according to predefined attachment and detachment rates (resp. ω_a and ω_d). The whole set of rules is summarized in figure 1.

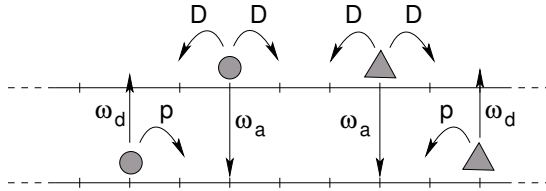


Fig. 1. Sketchy representation of the evolution rules. The two types of motors are represented respectively with disks and triangles. They move in opposite directions, but with the same rate p , on the lower lane representing the microtubule, on which there is an exclusion principle. Motors hop in both directions and without any exclusion constraint, on the upper lane representing the space around the microtubule in which diffusion can take place.

3 Bidirectional Traffic on a Static Filament Surrounded by a Confined Diffusive Reservoir

In [2], M. Ebbinghaus and L. Santen have shown that for the model described in the previous section, a transition to jamming occurs beyond a certain number of particles (i.e., for a constant density, beyond a certain system size).

This jamming has two negative consequences:

- First, the flux of each of the species of motors tends to zero as the system size becomes large, i.e. transport becomes very inefficient.
- Second, the remaining small flux depends on the system size - and not on the density of motors. Indeed, the jam that is formed involves a macroscopic fraction of the total number of motors present in the system - a quantity which is itself proportional to the system size.

This second point means that in large systems such as axons, jamming would indeed lead to a vanishing flux. Open systems, where the input flow of motors is not limited, and thus where the total number of motors is virtually infinite, would also lead to a vanishing flux.

Fig. 2 illustrates how the flux is vanishing for an increasing attachment rate ω_a - i.e. when more particles are sent on the processive filament. This transition

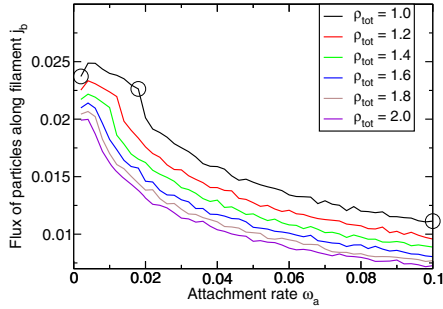


Fig. 2. Flux of positive particles on the microtubule lane, as a function of the attachment rate ω_a , for a fixed system size $L = 1000$, for $\omega_d = 0.01$, $D = 0.1$, $p = 1$, and for different total numbers of particles. The circles indicate from which simulations the snapshots of fig. 3 were taken.

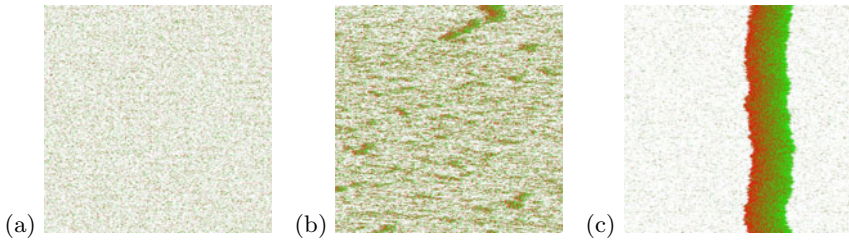


Fig. 3. Spatio-temporal snapshots in the various phases when ω_a is increased, for a fixed number of particles and a fixed system size $L = 1000$, for $D = 0.1$, $p = 1$, and $\omega_d = 0.01$. Green and red points indicate locations of positive and negative motors. (a) Free flow phase, no clusters are seen. (b) Transient clusters appear in the vicinity of the transition. (c) One large cluster inhibits processive motion on the filament. The corresponding points in fig. 2 are indicated as circles.

occurs for smaller value of ω_a when the system size increases. Around the jamming transition, as shown on the snapshots of fig. 3, some short lived clusters start to be formed. And beyond, a single large cluster containing a macroscopic fraction of the motors inhibits directed motion.

As seen in fig. 4, the density of the large cluster increases when ω_a increases (or, equivalently, when the system size increases). Indeed, there is then a strong correlation of the densities on both lanes, i.e. the density in the diffusive reservoir becomes high above the jam. As a consequence, the small holes in the jam are filled by attaching particles. As shown in 2, the number of attached particles grows linearly with the system size, while the length of the cluster grows sub-linearly.

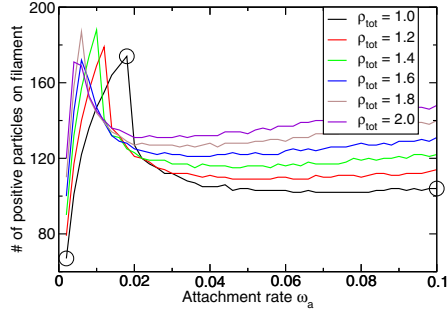


Fig. 4. Number of positive particles as a function of the attachment rate ω_a , for a fixed system size $L = 1000$, for $\omega_d = 0.01$, $D = 0.1$, $p = 1$, and for different total numbers of particles. The circles indicate from which simulations the snapshots of fig. 3 were taken.

4 Relaxation of the Confinement Constraint in the Case of a Static Filament

We have seen in the previous section that the explicit representation of the diffusive space as a unique line parallel to (i.e. close to) the microtubule was leading to a jamming transition. This would hold if the microtubule is placed in a crowded environment, which we believe to be the case in the cell - in particular, microtubules are never unique in the cell, they rather form some intricate bundles.

However, in different types of cells, the density of filaments can differ quite a bit. So we would like to study the effects of a larger diffusive volume on the general behavior. In the limit of no confinement at all, there would be almost no correlations between the motors detachment and attachment positions. As a consequence, jamming does not occur, as emerging jams are dissolved through evaporation in the environment (at least for a large enough detachment rate).

In this section, we consider an intermediate model that allows to crossover from one behavior to the other. We still consider the two lane model of section 2, but now the upper lane represents a diffusive reservoir with a certain depth. With this larger reservoir, we expect that after detaching from the microtubule, the motors will diffuse in the reservoir before coming again close to the microtubule - and eventually attaching. More precisely, if the reservoir is a cylinder with radius R , the average first return time to the center of the cylinder would scale as R^2 . We take this effect into account by taking an affective attachment rate $\tilde{\omega}_a = \omega_a/R^2$. On the other hand, assuming a constant average density of particles, the total number of particles in the system would scale as R^2 too. Actually we took for the total number of particles $N = (1+R^2)\rho_0$, assuming that the central line of

the cylinder represents at the same time the microtubule filament and the close diffusive neighborhood of the filament.

We should notice that the detachment rates control the length of a typical path on the microtubule, while the geometry around the microtubule influences the effective attachment rate.

In the limit of a vanishing attachment rate, and an infinite density of motors, we recover the models which were considering an infinite reservoir of particles, and no jamming transition is expected to occur for a realistic value of the detachment rate. On the other hand, for the value of ω_d considered in the previous section, we have seen that a jamming transition was occurring for large enough systems. Now we explore the crossover between the two behaviors.

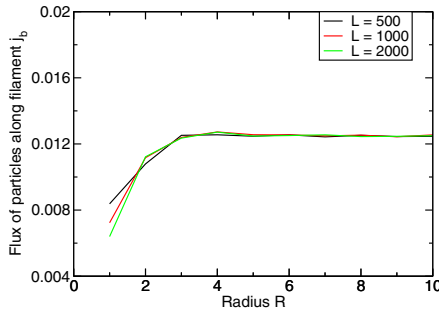


Fig. 5. Flux of positive particles on the microtubule lane, as a function of the radius R of the cylindrical reservoir, for $\omega_d = 0.01$, $\rho_0 = 0.5$, $D = 0.1$, $p = 1$, and for different system sizes

Fig. 5 shows indeed a crossover from a system size dependent state (under confinement) to a density dependent state (large reservoir). For small radius R , the density in the reservoir is affected by the state of the microtubule. Beyond a certain value of the radius R , the variations due to the microtubule become negligible in front of the large total number of particles in the reservoir, and an asymptotic regime is reached.

In this asymptotic regime, there is no macroscopic jam and as we said, flux is density dependent. However, there are still some small jams forming along the microtubule, and the flux values that are reached are not very high (compare with the next section for example). Thus, for a given density, the flux improvement due to the dynamics of the filament (see next section) can be much greater than the one due to non-confinement.

In nature, it is probable that there exists the whole range of situations, from confined to non-confined.

5 Bidirectional Traffic on a Dynamic Filament Surrounded by a Confined Diffusive Reservoir

If we consider the confined case, we have seen that bidirectional motion leads to jamming for systems of large size.

It turns out that microtubules are not stable structures; in fact, their plus end exhibits alternating phases of polymerization and depolymerization. According to one of the suggested scenarios, during polymerization, the plus end of the microtubule is covered by a GTP cap. As long as the cap is present, polymerization is favored. But it may happen that the GTP cap hydrolyzes to GDP+Pi, and then the plus end becomes unstable and depolymerizes rapidly. This event is called a catastrophe. Some mechanism, called *rescue*, allows to stop the catastrophe. It occurs stochastically and may prevent the microtubule from entirely depolymerizing. For small cells, where the microtubules cannot be very long, it is not rare that a microtubule is completely destroyed by depolymerization. By contrast, in mature axons, where microtubules are believed to measure a few hundreds microns, depolymerization is in general localized around the tip of the microtubule. Anyhow, the dynamics of the microtubules occur on time scales which are similar to those involve in jam formation - and thus a coupling is likely to take place. An illustration of the dynamics of the filaments can be found in the videos given as supplementary material of [7]. The polymerizing ends are made visible through fluorescence.

At this stage, we considered much simplified filament dynamics, in order to illustrate how the jamming transition can be hindered. The three types of dynamics that we considered were the following

- D1 : some sites of the microtubule are suppressed at random with rate k_d and restored with rate k_p .
- D2 : same rule, except that a site is eliminated with rate k_d *only* if it is occupied (this could happen for example if the motors were inducing a strain on the filament).
- D3 : regularly spaced holes in the lower lane propagate synchronously but stochastically (oversimplification of the so-called treadmilling of filaments).

We must define how the filament dynamics affects the motors (see fig. 6 for a sketchy representation). If a motor arrives at the end of a microtubule, i.e. if the motor wants to hop on a missing microtubule site, then it goes in the diffusive lane instead. If a microtubule site depolymerizes while there is a motor on it, the motor is also sent to the diffusive reservoir. As a result, the effective detachment rate is increased compared to the case of the static filament.

In [1], we have presented how a density dependent flux is recovered above a certain depolymerization rate k_d for dynamics D1 and D2, and above a certain density of holes for dynamics D3. As shown on Fig. 7, a maximum flux is obtained for an optimum compromise between a too low depolymerization rate (for which jams are not dissolved) and a too high one (for which there is not enough microtubule left to allow for efficient processive motion). The maximum flux

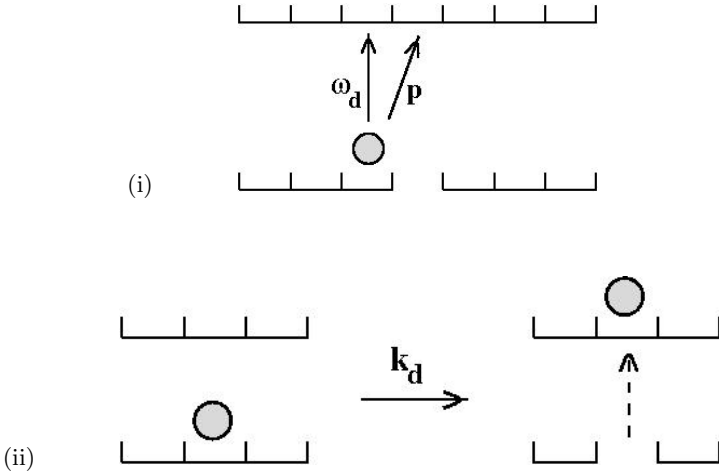


Fig. 6. Sketchy representation of the evolution rules (i) when a motor arrives on a missing microtubule segment; or (ii) when a microtubule segment on which a motor is present depolymerizes. Again, the diffusive lane is represented at the upper lane, and the microtubule as the lower lane.

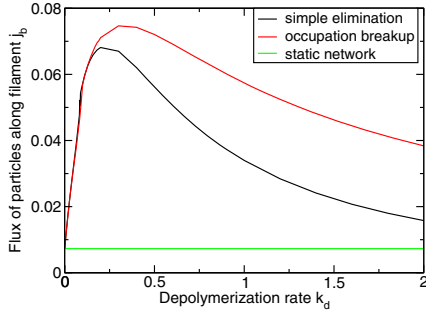


Fig. 7. Flux of positive particles on the microtubule lane, as a function of the depolymerization rate k_d . Other parameters are fixed to $\omega_a = 0.33$, $\omega_d = 0.02$, $D = 0.33$, $p = 1$, $k_p = 1$, and the system size $L = 1000$. The green, black, and red curves correspond respectively to a static filament, a filament undergoing dynamics (D1), and (D2). Note that the attachment rate considered here are much larger than in the figures of section 3, i.e. flow on the static filament is completely jammed.

values that are obtained are of the same order for the three types of dynamics (D1 and D2 shown on fig. 7).

As a summary, we find that, for quite different types of dynamics of the microtubule network, and for a large range of parameters:

- first, flux is increased, up to about 1/3 of the value it would have if oppositely motors would not “see” each other.

- second, flux now depends only on the density of motors, and not on the system size (at least beyond a certain system size threshold).

This type of mechanism could thus allow to recover efficient transport also in large systems, where the flux breakdown was expected to be the most severe in the static case.

6 Discussion and Conclusion

We have shown that a simple model of bidirectional transport leads to vanishing flux, and that this effect should be worse in large systems such as axons, as the remaining flux decreases with the system size.

It is an open question to know how nature turns this difficulty around. One possibility would be that motors interact in such a way that different species of motors would walk on different tracks [3]. But there is no experimental evidence yet of interactions that would be strong enough to lead to such segregation. This seems to be all the more difficult that motors are attached to quite large cargos, and thus cannot come close to each other, while the interactions that were pointed out between kinesins are rather short range.

Thus, though this scenario cannot be ruled out, it seems interesting to explore alternative possibilities. Here we have evidenced a counter intuitive effect, namely that suppressing some parts of the “road” can improve transport. Indeed, the dynamics of the network can prevent jam formation. As a consequence, larger values of the flux are obtained, and these values depend only on the density of motors - and not on the system size.

There would be a need for a better understanding of the microtubule network dynamics in particular in the axons. Various studies have shown that during the axon growth, microtubules were quite short and highly dynamic. It is less obvious how important this effect would be in mature axons. The importance of the steric effects due to the cargos needs also to be explored.

Acknowledgments. M.E. would like to thank the DFG Research Training Group GRK 1276 for financial support.

References

1. Ebbinghaus, M., Appert-Rolland, C., Santen, L.: Bidirectional transport on dynamic networks (2010) (submitted), arXiv:0912.3658
2. Ebbinghaus, M., Santen, L.: A model for bidirectional traffic of cytoskeletal motors. *J. Stat. Mech.*, P03030 (2009)
3. Klumpp, S., Lipowsky, R.: Phase transitions in systems with two species of molecular motors. *Europhys. Lett.* 66, 90–96 (2004)
4. Lipowsky, R., Klumpp, S., Nieuwenhuizen, T.M.: Random walks of cytoskeletal motors in open and closed compartments. *Phys. Rev. Lett.* 87, 108101 (2001)
5. Nishinari, K., Okada, Y., Schadschneider, A., Chowdhury, D.: Intracellular transport of single-headed molecular motors KIF1A. *Phys. Rev. Lett.* 95, 118101 (2005)

6. Parmeggiani, A., Franosch, T., Frey, E.: Phase coexistence in driven one dimensional transport. *Phys. Rev. Lett.* 90, 086601 (2003)
7. Shemesh, O.A., Erez, H., Ginzburg, I., Spira, M.E.: Tau-induced traffic jams reflect organelles accumulation at points of microtubule polar mismatching. *Traffic* 9, 458–471 (2008)
8. Stokin, G.B., Lillo, C., Falzone, T.L., Brusch, R.G., Rockenstein, E., Mount, S.L., Raman, R., Davies, P., Masliah, E., Williams, D.S., Goldstein, L.S.B.: Axonopathy and transport deficits early in the pathogenesis of alzheimer's disease. *Science* 307, 1282 (2005)
9. Tailleur, J., Evans, M., Kafri, Y.: Nonequilibrium phase transitions in the extraction of membrane tubes by molecular motors. *Phys. Rev. Lett.* 102, 118109 (2009)

Cellular Automata for a Traffic Roundabout

Ding-wei Huang

Department of Physics, Chung Yuan Christian University, Chung-li, Taiwan

Abstract. We propose a cellular automaton model for a typical traffic roundabout to regulate traffic flow from four different directions. In the four-way symmetrical case, the number of parameters reduce from sixteen to four. As these four parameters change, we observe three distinct traffic phases: free flow, congestion, and gridlock. As the inflow increases, free flow transits to congestion when the interweaving traffic is light. When the interweaving traffic is heavy, free flow transits directly to gridlock instead. The traffic interweave is characterized by a new parameter X . We present both numerical simulations and analytical discussions.

Keywords: Traffic Flow, Congestion, Gridlock, Intersection.

1 Introduction

Traffic-related problem can be one of the major challenges in modern society. With the mathematical tools of differential equations, traffic dynamics has been analogized to hydrodynamics via partial differential equations and to force dynamics via ordinary differential equations [1]. More recently, cellular automata become another useful tool to explore the traffic dynamics [2,3]. For the traffic flow on a simple and homogeneous roadway, different approaches can all be applied to reach more or less the same result. However, cellular automata can be much more convenient to use when the roadways become more and more complicated.

At the most basic level, traffic dynamics is often discussed on a homogeneous roadway. To go next step, it becomes necessary to consider the road intersection. At an intersection, the limited space has to be shared by vehicles from different directions. Various schemes have been used to resolve the obvious traffic conflicts. One type of schemes requires a vehicle to come to a full stop, e.g. stop sign and traffic signal. The other type of schemes tries to avoid the full stop of vehicles, e.g. traffic circle, rotary, and roundabout [4]. In this work, we propose a cellular automaton model to analyze the traffic flow at a typical roundabout. We simply adopt the characteristics of a roundabout: (a) traffic entering the roundabout must yield the right-of-way to traffic already in the circle; (b) no lane changes occur within the circle; (c) vehicle speeds are low. Compared to traffic signal, the advantage of a roundabout seems to be that vehicles are able to keep on moving without a full stop. However, the roundabout cannot guarantee the fluidity of traffic at the intersection. We will also analyze the gridlock at the roundabout. We present the model in the next section, followed by the results of numerical simulations. Some analytical properties will be discussed later.

2 Model

The system configuration is shown in Fig. 1 (left), where eight single-lane roadways are connected to a single-lane ring. Traffic toward the ring is labeled by an odd number; traffic away from the ring is labeled by an even number. In the ring, vehicles move counter-clockwise. The number of vehicles is not conserved in the system. Vehicles move into the system through the odd-numbered boundaries, After travelled various distances in the ring, vehicles move out of the system through the even-numbered boundaries. The roundabout can be taken as four connected T-shaped intersections [5] shown in Fig. 1 (right). In the model, roadways are divided into discrete cells. Each cell can be either empty or occupied by one vehicle. The motion is in one direction only, which is shown by the gray-bold arrow in Fig. 1. The vehicle on each cell has a unique direction to follow. If an empty cell is available, the vehicle will move forward in the next time step. The only exception is the shaded cell shown in Fig. 1 (right), which locates right at the intersection. On the shaded cell, the vehicle may turn right to exit the ring, or move straight forward to remain in the ring. These two choices are determined by a quench variable assigned to each vehicle while entering the system. It can be reasonable to assume that vehicles approaching the roundabout have their own predestined journeys.

We adopt the stochastic boundary condition for vehicles to enter the system. At the odd-numbered boundaries, if the first cell is empty, a new vehicle is added stochastically with a finite probability α_i . Basically the parameter α_i controls the inflow through boundary i . The destiny of this newly added vehicle is also selected stochastically. With a finite probability P_{ij} , the vehicle enters the system through boundary i and exits through boundary j , where $i = 1, 3, 5, 7$, and $j = 2, 4, 6, 8$. At the even boundaries, vehicles will leave the system freely. The dynamics is described by the Asymmetric Simple Exclusion Process (ASEP)

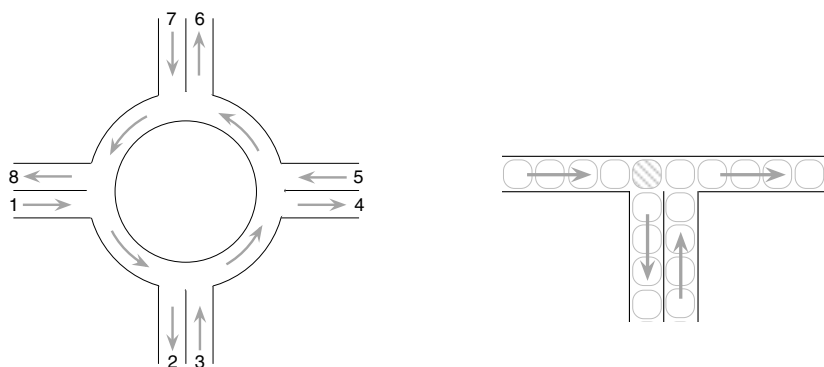


Fig. 1. Left. Configuration of a roundabout, which consists of four T-shaped intersections. **Right.** Configuration of a T-shaped intersection. Traffic direction on each roadway is indicated by the gray-bold arrow.

with parallel update. Whenever the next cell is empty, vehicle moves forward. This simple rule is applied to all vehicles synchronously. A traffic conflict can be expected at the intersection shown in Fig. 1 (right). Without further regulations, two vehicles might move into the cell next to the shaded cell simultaneously. To avoid such a conflict, we adopt a conventional regulation: entering vehicle should yield to vehicle in the ring. The vehicle occupied the shaded cell has the right-of-way to move forward. In this model, the traffic conditions are specified by the injection α_i and the distribution P_{ij} , where $i = 1, 3, 5, 7$, and $j = 2, 4, 6, 8$. With the four normalizations for P_{ij} , there are sixteen parameters in total.

3 Numerical Results

Congestion emerges whenever traffic demand exceeds roadway capacity. We analyze the emergent patterns of congestion due to the increase of inflow. In this section, we further assume the four-way symmetry, i.e. $P_{12} = P_{34} = P_{56} = P_{78}$, $P_{14} = P_{36} = P_{58} = P_{72}$, $P_{16} = P_{38} = P_{52} = P_{74}$, $P_{18} = P_{32} = P_{54} = P_{76}$, and $\alpha_1 = \alpha_3 = \alpha_5 = \alpha_7$. The twenty parameters can be reduced to P_{12} , P_{14} , P_{16} , P_{18} , and α_1 , with a constraint $P_{12} + P_{14} + P_{16} + P_{18} = 1$. As these parameters change, the system displays three different phases: free flow, congestion, and gridlock. In the free flow, all vehicles are able to move freely. The incoming roadways and the outgoing roadways have the same vehicular density; while the density in the ring assumes a higher value owing to the traffic interweave. In the congestion, the traffic in the ring becomes congested. The incoming roadways are also congested and have a higher density than the density in the ring; while the outgoing roadways remain free flowing and have a lower density. In the gridlock, both the ring and the incoming roadways are jam-packed with vehicles. No further vehicles can enter the system. And the outgoing roadways become empty.

As the injection α_1 increases, with fixed distribution P_{ij} , we observe only one transition. The free flow transits into either congestion or gridlock depending on the distribution P_{ij} . In Fig. 2 (left), we show a typical example of traffic response when the traffic flow from different directions does not interweave too much. With the parameters $P_{12} = 0.9$, $P_{14} = 0.1$, and $P_{16} = P_{18} = 0$, we prescribe 90% of incoming vehicles to travel a quarter of the ring and then to make a right turn at the first exit to leave the system. The remaining 10% of vehicles will travel half of the ring and then leave the system at the second exit. The congestion emerges as the conventional phase transition. Before the transition, vehicular densities increase with the increase of injection. After the transition, vehicular densities saturate and become independent of the injection. When the traffic flow interweaves heavily, the gridlock appears swiftly. A typical example is shown in Fig. 2 (right). With the parameters $P_{12} = P_{14} = 0.1$, $P_{16} = 0.8$, and $P_{18} = 0$, we prescribe 80% of incoming vehicles to travel three-quarters of the ring and to leave the system at the third exit. At the first exit, 10% of vehicles leave the system; another 10% of vehicles leave at the second exit. Before the transition, free flow can be observed. After the transition, the density on the

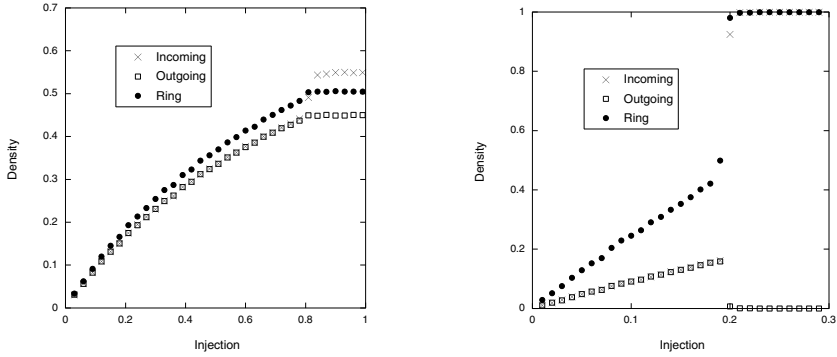


Fig. 2. Vehicular density as a function of injection. **Left.** Free flow transits to congestion when the interweaving traffic is light ($X = 1.1$). **Right.** Free flow transits to gridlock when the interweaving traffic is heavy ($X = 2.7$).

incoming roadways and the ring shoots to one. As a consequence, the density on the outgoing roadways drops to zero.

4 Discussions

In the free flow, traffic dynamics is controlled by the injection α . The vehicular densities can be written as

$$\rho_i = \frac{\alpha}{1 + \alpha} = \rho_o = \frac{1}{X} \rho_r \quad , \quad (1)$$

where ρ_i , ρ_o , and ρ_r denote respectively the densities of the incoming roadway, the outgoing roadway, and the ring. The parameter X characterizes the interweave traffic in the ring and is related to the distribution P_{1j} as

$$X = 4 - 3P_{12} - 2P_{14} - P_{16} \quad , \quad (2)$$

where the parameter has a range of $1 < X < 4$. As α increases, all the densities increase accordingly. The phase transition point can be determined when the density in the ring reaches the maximum, i.e. $\rho_r = \frac{1}{2}$. The critical injection can be obtained as

$$\alpha = \frac{1}{2X - 1} \quad . \quad (3)$$

As X increases, the interweave traffic becomes heavier and the phase transition appears at a lower injection.

In the congestion, traffic flow is independent of α . The relations among the densities become

$$\rho_i = 1 - \rho_o = \frac{1}{X} \rho_r + \left(1 - \frac{1}{X}\right) \quad , \quad (4)$$

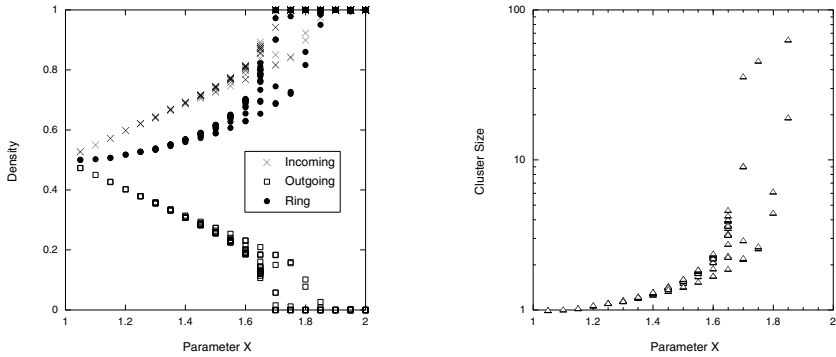


Fig. 3. **Left.** Vehicular density as a function of parameter X . **Right.** Cluster size as a function of parameter X . The parameters are $\alpha_1 = 1$ and various P_{1j} .

where $\rho_0 = (1 - \rho_r)/X$. We find that the saturated densities are mainly controlled by the parameter X . In Fig. 3 (left), data from various P_{1j} collapse into the same curve. As X increases, the congestion becomes severe and the traffic flow diminishes. The densities increase both in the incoming roadway and in the ring; and the density in the outgoing roadway decrease accordingly. Around a critical value of $X = 1.7$, the gridlock begins to emerge and the traffic flow drops to zero. We observe that the decrease of traffic flow can be attributed to the increase of vehicular cluster. In the free flow, all the vehicles are separated with at least one empty cell, i.e. the cluster size is one. In the congestion, the cluster size begins to increase, see Fig. 3 (right). When the cluster size reaches the order of the ring segment, gridlock becomes inevitable. It would be interesting to further develop a mean field theory for the critical behavior of congestion. The work is under progress for the cases where the symmetry assumption for parameters (P_{ij}, α_k) is relaxed. We expect that this simple system of roundabout will present more complicated and interesting behaviors in such asymmetrical cases.

References

1. Kerner, B.S.: The Physics of Traffic: Empirical Freeway Pattern Features, Engineering Applications, and Theory. Springer, Berlin (2004)
2. Chowdhury, D., Santen, L., Schadschneider, A.: Statistical Physics of Vehicular Traffic and Some Related Systems. *Phys. Rep.* 329, 199–329 (2000)
3. Maerivoet, S., De Moor, B.: Cellular Automata Models of Road Traffic. *Phys. Rep.* 419, 1–64 (2005)
4. Huang, D.W.: Phase Diagram in Traffic Dynamics. In: Moreno, J.S. (ed.) *Progress in Statistical Mechanics Research*, pp. 373–411. Nova Science, New York (2008)
5. Huang, D.W.: Complete Traffic Patterns around a T-shaped Intersection. *Int. J. Mod. Phys. C* 21, 189–204 (2010)

Cellular Automata for a Cyclic Bus

Ding-wei Huang and Wei-neng Huang

Department of Physics, Chung Yuan Christian University, Chung-li, Taiwan

Abstract. We propose a simple cellular automaton model to study the dynamics of a cyclic bus. The nontrivial fluctuations are prescribed by the stochastic moving of bus interacting with the stochastic arrival of passengers. As passengers increase, the bus schedule shows a clear transition. Both numerical and analytical results are presented. The divergence of bus schedule can be taken as an analogy to the gridlock of 4-way traffic. We also comment on the strategy to keep a stable schedule.

Keywords: Traffic Flow, Congestion.

1 Introduction

Recently, traffic dynamics has attracted much attention from physicists [1,2,3]. One of the research interests concerns the intrinsic fluctuations of the dynamics. For a highway system, the dynamics can be simple and deterministic; each vehicle follows the preceding smoothly down the road. However, as vehicular density increases, a small fluctuation in one of the headways will lead to instability of the whole system, then the congestion emerges inevitably without any specific causes. For the urban traffic, the situations can be much more complicated. Further fluctuations should be considered, e.g., traffic from different directions [4], operation of traffic lights [5], and the interaction with pedestrians [6]. It is interesting to note that the fluctuations can be nontrivial even in the case of a single vehicle [7,8]. In this paper, we will focus on such a case where a cycling bus moves along a closed route and interacts with the passengers waiting to get on the bus.

2 Bus Route Model

The travelling bus is taken as a particle hopping along a discrete lattice periodically. Consider a cyclic bus route consisting of M stops; at each bus stop, the passengers arrive at a rate γ . As a bus hops along the route, the hopping rate p is strongly influenced by the number of passengers N waiting at each stop. When N increases, p decreases accordingly to prescribe a delayed bus. In the original bus route model [9], there was no such dependence; later, a linear dependence was considered [10]. Subsequently, a much stronger dependence was proposed [8]. Here, we adopt a simple quadratic form,

$$p = \frac{1}{1 + aN^2} . \quad (1)$$

As a naive scaling of $(a \cdot \gamma^2)$ is expected, we assume $a = 1$ without loss of generality. In the model, there are only two parameters M and γ . When there is no passenger to delay the hopping, $\gamma = 0$, the bus completes the route at a fixed schedule $\Delta T = M$. As γ increases, ΔT is expected to increase accordingly. When a bus is delayed, there would be more passengers waiting at the bus stop, and as more passengers are accumulated, the bus would be further delayed. Thus, an instability can be expected as γ increases.

In the mean-field approximation, the stochasticity is suppressed. With a schedule of ΔT_i , the average number of passengers waiting at each bus stop is $(\gamma \cdot \Delta T_i)$ and the next recurrence bus would spend an average time $[1 + (\gamma \cdot \Delta T_i)^2]$ there. Thus we obtain the following nonlinear map of a single variable ΔT ,

$$\Delta T_{i+1} = M [1 + (\gamma \cdot \Delta T_i)^2] \quad , \tag{2}$$

where the subscripts of ΔT denote the recurrence index. In this mean-field theory, ΔT diverges as γ increases. The critical value can be obtained as

$$\gamma > \frac{1}{2M} \quad . \tag{3}$$

With a small γ , the stable bus schedule is as following

$$\Delta T = \frac{1 - \sqrt{1 - 4\gamma^2 M^2}}{2\gamma^2 M} \quad . \tag{4}$$

A scaling of $(M \cdot \gamma)$ is also observed. At the limit of $\gamma \rightarrow 0$, the above analytic formula reproduces the fixed point of $\Delta T = M$; at the other limit of $\gamma \rightarrow 1/(2M)$, ΔT approaches its maximum of $(2M)$. In between these two limits, ΔT increases smoothly with the increase of γ .

In the cellular automaton simulations, the averaged ΔT increases with the increase of γ much faster than the prediction of mean-field theory. The critical value appears to be much less than the mean-field prediction at $\gamma = 1/(2M)$, see Fig. 1 (left). For each recurrence i , wide fluctuations of ΔT_i can be noticed. In Fig. 1 (right), we plot the probability for a bus to have a stable schedule. With a smaller γ , the bus recurs stably; with a larger γ , the schedule diverges easily.

To look into more details, we plot the probability distributions of ΔT at various γ , see Fig. 2 (left). With the above mean-field results, we would naively expect a simple distribution prescribing an increasing ΔT with an increase γ . Instead, we observe an interesting distribution still dominated by the fixed schedule of $\Delta T = M$. As γ increases, the probability to keep the schedule decreases exponentially. Only when the dominant peak at $\Delta T = M$ subsides, did the secondary distribution at $\Delta T > M$ become obvious. For the broad distribution of the secondary structure, the mean of ΔT shifts toward larger values as γ increases.

The probability to keep the schedule can be expressed as

$$\left[\sum_{N=0}^M C_N^M (1 - \gamma)^{M-N} \gamma^N \cdot \frac{1}{1 + N^2} \right]^M \sim \exp \left(-\frac{M^2}{2} \gamma \right) \quad . \tag{5}$$

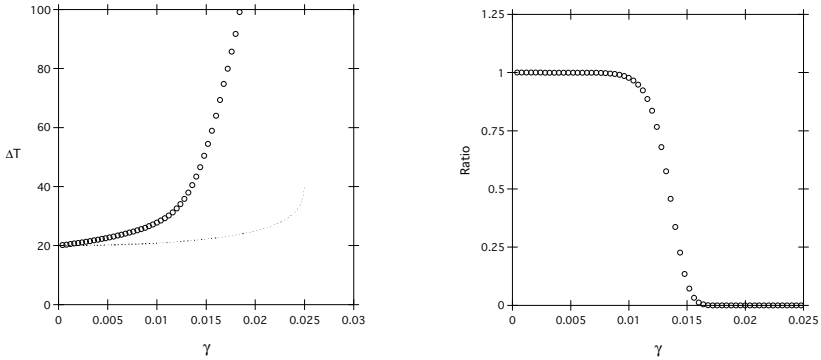


Fig. 1. Left. Averaged recurrence schedule ΔT as a function of γ , where $M = 20$. The dotted line shows the mean-field prediction, which terminates at $\gamma = 1/(2M)$ with a maximum $\Delta T = 2M$. **Right.** Ratio of stable schedule as a function of γ , where $M = 20$. A stable schedule is defined as being able to recur at $i = 100$, where the divergence is taken as $\Delta T_i > 10M$.

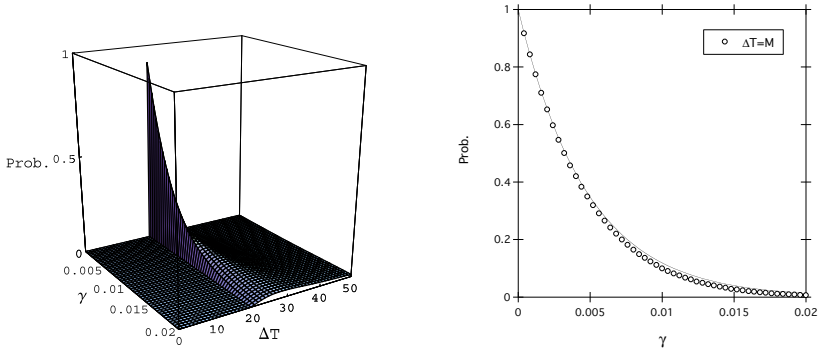


Fig. 2. Left. Probability distributions of ΔT at various γ , where $M = 20$. A dominant peak at $\Delta T = M$ superimposes on the secondary structure at $\Delta T > M$. **Right.** Decay of the primary peak at $\Delta T = M$. The solid line shows the analytical result of a simple exponential distribution, Eq. (5).

The binomial distribution represents the probability of N passengers waiting at a bus stop in a time span of M . With these passengers, the bus has a finite probability of keeping the schedule as prescribed in Eq. (4), and the same factor for each bus stop results in the power M . For a large M , the above analytic expression can be well approximated by a single exponential distribution. The numerical simulations can be fairly reproduced, especially for small γ .

3 Discussions

In this paper, we propose a cellular automaton model to study the dynamics of a cycling bus. The intrinsic fluctuations in the traffic dynamics are prescribed by the stochastic moving of bus coupled with the stochastic arrival of passengers. When the passengers increase, the schedule diverges suddenly, i.e., the bus was delayed indefinitely and unexpectedly. In reality, such phenomena can be found in the situations where the number of passengers waiting to get on the bus far surpasses the capacity of transportation. This is analogous to the gridlock appeared in an intersection where a few vehicles from different directions block each other and all the traffic is stopped indefinitely.

We also compare the results with previous finding based on the nonlinear maps, which prescribes an abrupt transition at $\gamma = 1/(2M)$. In the cellular automaton simulations, where the fluctuations were properly taken care of, we observe a smoother transition at a much small critical value. In the deterministic mean-field theory, the critical value was overestimated by a factor of two. As the fluctuations play an important role in traffic phenomena, the conjectures based on deterministic theory can be misleading. For example, a strategy was proposed to avoid the divergent schedule: by skipping a few stops, the bus will be able to keep a stable schedule [8]. As shown in this study, the stable schedule can only be reached by limiting the value $(M \cdot \gamma)$. With a fixed γ , the only option is to reduce the number of stops M . We point out that the conclusion in Ref. [8] was based on a unrealistic presumption that when the bus skips a stop, those passengers waiting at the bus stop disappear. We find that at a fixed γ , the only feasible strategy to stabilize the recurrent schedule is to add more buses to the route. It is well known that the same instability will lead these buses to bunch together as the passengers increase [9,11]. By instructing the bus drivers to skip a few stops will keep these buses more or less equal distanced, which would provide an effectively reduced M without presuming a reduced γ . Thus the stable scheme can be restored by the strategy of adding more buses in addition to skipping a few stops when necessary.

References

1. Maerivoet, S., De Moor, B.: Phys. Rep. 419, 1 (2005)
2. Chowdhury, D., Santen, L., Schadschneider, A.: Phys. Rep. 329, 199 (2000)
3. Helbing, D.: Rev. Mod. Phys. 73, 1067 (2001)
4. Biham, O., Middleton, A.A., Levine, D.: Phys. Rev. A46, R6124 (1992)
5. Huang, D.W., Huang, W.N.: Phys. Rev. E67, 056124 (2003)
6. Jiang, R., Helbing, D., Shukla, P.K., Wu, Q.S.: Physica A368, 568 (2006)
7. Nagatani, T.: Physica A297, 260 (2001)
8. Nagatani, T.: Physica A316, 637 (2002)
9. O'Loan, O.J., Evans, M.R., Cates, M.E.: Phys. Rev. E58, 1404 (1998)
10. Nagatani, T.: Physica A312, 251 (2002)
11. Nagatani, T.: Physica A305, 629 (2002)

Dynamics of a Tagged Particle in the Asymmetric Exclusion Process with Particlewise Disorder

Takashi Imamura

Research Center for Advanced Science and Technology, The University of Tokyo,
Komaba 4-6-1, Meguro-ku, Tokyo 153-8902, Japan
timamura@iis.u-tokyo.ac.jp

Abstract. We consider the one-dimensional totally asymmetric asymmetric simple exclusion process. We particularly concentrate on the case where each hopping rate depends on each particle. In the step initial condition, where all sites from the left of some site are occupied and all other sites are empty, we discuss a dynamics of a particular particle (tagged particle). We show that the position fluctuation of the tagged particle is equivalent to the largest eigenvalue fluctuation of the Gaussian Unitary Ensemble (GUE) in the random matrix theory.

1 Introduction

The asymmetric simple exclusion process (ASEP) is one of the most fundamental stochastic cellular automata. It is described by the following two rules. 1. Each particle hops to the right and left neighboring sites with different hopping rates (asymmetric diffusion effect). 2. If the target site is occupied by another particle, the particle cannot hop to this site (exclusion effect). Because of the interplay of these effects, the ASEP shows various interesting phenomena such as a boundary-induced phase transition, shock wave and so on [1, 2].

The remarkable feature of the ASEP is that the physical quantities such as current, density profile and so on can be calculated exactly. In particular, since the work of Derrida, Evans, Hakim and Pasquier [3], many studies on steady state properties of the ASEP and related models have been made [4].

On the other hand, it is also an interesting problem to understand the non-stationary properties of the ASEP such as a relaxation to a steady state. Recently Johansson [5] exactly analyzed the distribution function of the current in the ASEP under a non-stationary situation and found a surprising connection with the random matrix theory. Since then, the dynamics in the ASEP has been vigorously discussed [6].

In this paper, we discuss the dynamics of a particular particle (called a “tagged” particle) in the ASEP using the random matrix approach. In our previous study [7], we considered this problem in the case of a single hopping rate and the case where in an infinite number of “normal” particles, there are a few (a finite number of) “slow” particles whose hopping rate is smaller than that of

the normal particles. Here we are interested in the case where the particles have their own hopping rates. The purpose of this study is to understand how such an effect of particle-wise disorder affects the dynamics of a tagged particle.

The paper is organized as follows. In the next section we describe the model and summarize the tagged particle properties studied in [7]. In Sec. 3 we present our main result. The conclusion and discussion are given in Sec. 4.

2 Tagged Particle in the ASEP

2.1 Model

In this paper we consider the one-dimensional totally asymmetric simple exclusion process(TASEP) with parallel update rule. We consider particles on the one-dimensional infinite lattice as illustrated in Fig.1(a). Each particle moves stochastically obeying the following rules (Fig.1(a)):

1. During each time step between $t \in \{0, 1, \dots\}$ and $t + 1$, the particle labeled i moves to the right neighboring site with probability $1 - q_i$ or stays with probability q_i if the right neighboring site is empty. (The particles do not go to the left neighboring sites.)
2. If the right neighboring site is occupied by the particle labeled $j - 1$, the j th particle cannot move to this site.

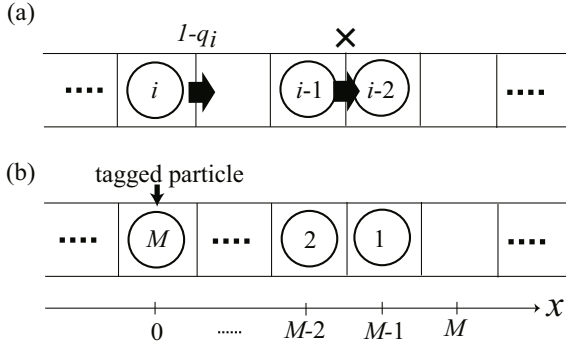


Fig. 1. (a) A schematic diagram of the TASEP. (b) The step initial condition.

2.2 Tagged Particle Dynamics

Under the step initial condition, where all sites from the left of a certain site are occupied by the particle and all right sites are empty as depicted in Fig. 1(b), we consider the dynamics of the particle labeled M .

Let $X(t)$ be the position of the M th particle. (Here we set the site occupied by this particle at $t = 0$ as the origin as depicted in Fig 1(b).) In this paper we focus on this quantity.

In [7], we obtained the multi-time probability distribution of $X(t)$ as follows. When $t \geq M$, we have

$$\text{Prob}(X(t_1) \geq \ell_1, X(t_2) \geq \ell_2, \dots, X(t_m) \geq \ell_m) = \det(1 + Kg). \tag{2.1}$$

Here $\det(1 + Kg)$ is the Fredholm determinant defined by

$$\begin{aligned} \det(1 + Kg) &= \sum_{k=0}^{\infty} \frac{1}{k!} \sum_{n_1=1}^m \sum_{x_1=-\infty}^{\infty} \cdots \sum_{n_k=1}^m \sum_{x_k=-\infty}^{\infty} g(t_{n_1}; x_1) \cdots g(t_{n_k}; x_k) \\ &\quad \times \det(K(t_{n_l}, x_l; t_{n_{l'}}, x_{l'}))_{l, l'=1}^k, \end{aligned} \tag{2.2}$$

where

$$\begin{aligned} g(t_n; x) &= -\chi_{(t_n - M + 1 - \ell_n, \infty)}(x), \quad (n = 1, 2, \dots, m), \\ \chi_{(a,b)}(x) &= \begin{cases} 1, & \text{if } a < x < b, \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{2.3}$$

and the kernel $K(t_1, x_1; t_2, x_2)$ is given by

$$\begin{aligned} K(t_1, x_1; t_2, x_2) &= \tilde{K}(t_1, x_1; t_2, x_2) - \phi_{t_1, t_2}(x_1, x_2), \\ \tilde{K}(t_1, x_1; t_2, x_2) &= \frac{1}{(2\pi i)^2} \int_{C_{R_1}} \frac{dz_1}{z_1} \int_{C_{R_2}} \frac{dz_2}{z_2} \frac{z_1}{z_1 - z_2} \frac{(1 + 1/z_2)^{t_2 - M + 1}}{(1 + 1/z_1)^{t_1 - M + 1}} \prod_{i=1}^M \frac{1 - q_i - q_i z_2}{1 - q_i - q_i z_1} \frac{z_2^{x_2}}{z_1^{x_1}}, \end{aligned} \tag{2.4}$$

$$\phi_{t_1, t_2}(x_1, x_2) = \begin{cases} \frac{1}{2\pi i} \int_{C_1} \frac{dz}{z} \left(1 + \frac{1}{z}\right)^{t_2 - t_1} z^{x_2 - x_1}, & t_1 < t_2, \\ 0, & t_1 \geq t_2. \end{cases} \tag{2.6}$$

Here C_R in (2.5) and (2.6) denotes a contour enclosing the origin anticlockwise with radius R and R_i ($i = 1, 2$) in (2.5) satisfy the conditions $R_2 < R_1$ and $1 < R_1 < (1 - q_i)/q_i$. The process characterized by the Fredholm determinant is called the Schur process [9,10,11].

Using this, we consider asymptotic behavior of $X(t)$ when both t and M go to infinity with the ratio t/M fixed. Here we assume that all hopping rates are the same,

$$q_1 = q_2 = \dots = q_M = q. \tag{2.7}$$

Let $x(\tau_i)$ and τ_i be the scaled position and scaled time of a tagged particle defined by

$$x(\tau_i) = \frac{X(t_i) - A(t_i/M)M}{D(u)M^{1/3}}, \tag{2.8}$$

$$t_i = uM + C(u)M^{2/3}\tau_i, \tag{2.9}$$

where

$$A(u) = (1 - q)u - (1 - 2q) - 2\sqrt{q(1 - q)(u - 1)}, \tag{2.10}$$

$$C(u) = 2(u - 1)^{\frac{5}{6}} \left(1 + \sqrt{\frac{1 - q}{q(u - 1)}} \right)^{\frac{1}{3}} \left(\sqrt{u - 1} - \sqrt{\frac{q}{1 - q}} \right)^{\frac{1}{3}}, \tag{2.11}$$

$$D(u) = (u - 1)^{\frac{1}{6}} q^{\frac{1}{2}} (1 - q)^{\frac{1}{2}} \left(1 + \sqrt{\frac{1 - q}{q(u - 1)}} \right)^{\frac{2}{3}} \left(\sqrt{u - 1} - \sqrt{\frac{q}{1 - q}} \right)^{\frac{2}{3}}. \tag{2.12}$$

Now let us confirm the meaning of $x(\tau)$. The solid line of Fig.2 shows the simulation of $X(t)$ when M is quite large ($M = 3000$). Note that due to the step initial condition, the particle cannot move until $t = M$. The dashed line indicates its average position $A(t/M)M$. Now we would like to understand the fluctuation of $X(t)$ around the average position. For this purpose, we fix the scaled time $u = t/M$ and focus on the dynamics when the time is around uM . We subtract the average position $A(t/M)M$ from $X(t)$ and scale the position axis (vertical axis) as $M^{1/3}$ and time axis (horizontal axis) as $M^{2/3}$. Then we can see the fluctuation of the tagged particle. We define this stochastic process as $x(\tau)$. (See Fig.2.) Note that the the set of exponent $(1/3, 2/3)$ is different from the case of the one-particle simple random walk $(1/2, 1)$. The hard-core repulsive interaction (the exclusion effect) causes the deviation. These exponents characterizes the one-dimensional Kardar-Parisi-Zhang (KPZ) universality class in non-equilibrium systems [8].

We obtained the following result on $x(\tau)$.

$$\lim_{M \rightarrow \infty} \text{Prob}(x(\tau_1) \geq s_1, x(\tau_2) \geq s_2, \dots, x(\tau_m) \geq s_m) = \det(1 + \mathcal{K}_2 \mathcal{G}). \tag{2.13}$$

Here the right hand side is the Fredholm determinant defined as

$$\begin{aligned} \det(1 + \mathcal{K}_2 \mathcal{G}) &= \sum_{k=0}^{\infty} \frac{1}{k!} \sum_{n_1=1}^m \int_{-\infty}^{\infty} d\xi_1 \cdots \sum_{n_k=1}^m \int_{-\infty}^{\infty} d\xi_k \mathcal{G}(\tau_{n_1}, \xi_1) \cdots \mathcal{G}(\tau_{n_k}, \xi_k) \\ &\quad \times \det(\mathcal{K}_2(\tau_{n_l}, \xi_l; \tau_{n_{l'}}, \xi_{l'}))_{l, l'=1}^k \end{aligned} \tag{2.14}$$

where $\mathcal{G}(\tau_j, \xi)$ ($j = 1, \dots, m$) are defined in terms of $\chi_{(a,b)}(x)$ [2,3], as

$$\mathcal{G}(\tau_j, \xi) = -\chi_{(s_j, \infty)}(\xi) \quad (j = 1, \dots, m), \tag{2.15}$$

and the kernel $\mathcal{K}_2(\tau_1, \xi_1; \tau_2, \xi_2)$ is given by

$$\mathcal{K}_2(\tau_1, \xi_1; \tau_2, \xi_2) = \begin{cases} \int_0^{\infty} d\lambda e^{-\lambda(\tau_1 - \tau_2)} \text{Ai}(\xi_1 + \lambda) \text{Ai}(\xi_2 + \lambda), & \tau_1 \geq \tau_2, \\ -\int_{-\infty}^0 d\lambda e^{-\lambda(\tau_1 - \tau_2)} \text{Ai}(\xi_1 + \lambda) \text{Ai}(\xi_2 + \lambda), & \tau_1 < \tau_2. \end{cases} \tag{2.16}$$

The stochastic process characterized by the Fredholm determinant with the kernel [2,16] is called the Airy process [12].

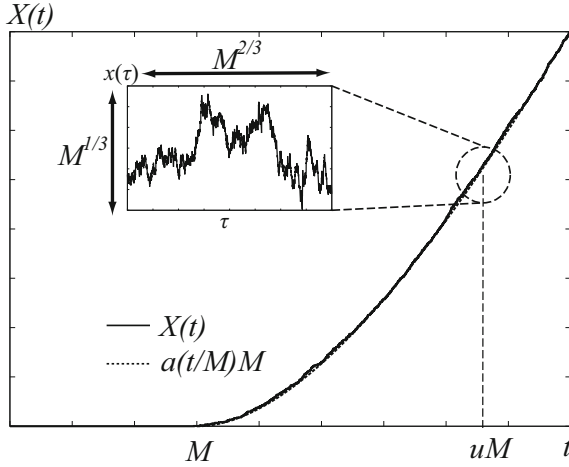


Fig. 2. The Monte-Carlo simulation of $X(t)$ ($M = 3000$)

The Airy process initially appears in the random matrix theory [13]. Let us consider the $N \times N$ hermitian matrix.

$$\begin{pmatrix} a_{11}(t) & a_{12}(t) + ib_{12}(t) & \cdots & a_{1N}(t) + ib_{1N}(t) \\ a_{12}(t) - ib_{12}(t) & a_{22}(t) & \cdots & a_{2N}(t) + ib_{2N}(t) \\ \vdots & \vdots & \ddots & \vdots \\ a_{1N}(t) - ib_{1N}(t) & a_{2N}(t) - ib_{2N}(t) & \cdots & a_{NN}(t) \end{pmatrix}, \quad (2.17)$$

where $a_{ij}(t)$, $b_{ij}(t)$ are described by the independent Brownian motions (Ornstein-Uhlenbeck processes). This random matrix model is called the GUE Dyson’s Brownian motion model. In particular, when we fix the time t , this is called the Gaussian Unitary Ensemble (GUE). In this matrix, we focus on the eigenvalues $\lambda_1(t) \geq \lambda_2(t) \geq \cdots \geq \lambda_N(t)$. Although the elements $a_{ij}(t)$, $b_{ij}(t)$ are independent random variables, the eigenvalues are not independent but correlated random variables with a repulsive interaction. We are especially interested in dynamics of the largest eigenvalue $\lambda_1(t)$. We scale it as $\bar{\lambda}_1(\tau) = (\lambda_1(t = N^{2/3}\tau) - \sqrt{2N})\sqrt{2N}^{1/6}$ where N represents the rank of the matrix (2.17). The scaled process $\bar{\lambda}_1(\tau)$ is described by the Airy process:

$$\lim_{N \rightarrow \infty} \text{Prob}(\bar{\lambda}_1(\tau_1) \leq s_1, \cdots, \bar{\lambda}_1(\tau_m) \leq s_m) = \det(1 + \mathcal{K}_2 \mathcal{G}). \quad (2.18)$$

When $m = 1$, the distribution is called the GUE Tracy-Widom distribution [14].

We find from (2.13) and (2.18) that in the TASEP with the step initial condition, the (scaled) dynamics of a tagged particle is equivalent to that of the largest eigenvalue in the GUE Dyson’s Brownian motion model.

3 ASEP with Particle-Wise Disorder

In the previous section, we consider the asymptotics of the tagged particle in the case of a single hopping rate (2.7). In this section, we discuss the case where the hopping rate depends on each particle.

Let $F(q)$ be the probability density function on $0 < q < 1$. In the step initial configuration (Fig.1(b)), we determine the hopping rate $q_i (i = 1, \dots, M)$ independently using the probability density function $F(q)$. (Once we generate them, we fix the values.) Under this initial condition, we consider the position fluctuation of the M th particle as in the previous section.

Before doing this, we focus on the dynamics of all M particles. Fig.3. shows the Monte-Carlo simulation with $M = 400$ and 4000 time steps. We consider the cases of two typical probability density function $F(q)$. The left figure represents the case of the unit density $F(q) = 2 \times \mathbf{1}_{(0,1/2)}(q)$ where $\mathbf{1}_{(a,b)}(q) = 1$ for $a < q < b$ and $= 0$ otherwise. The right figure corresponds to the case $F(q) = 6(1 - 2q^2)\mathbf{1}_{(0,1/2)}(q)$. The vertical axis represents the position of the particles and horizontal one represents the time. There are two time scales τ_1 and τ_2 in these figures

$$\tau_1 = \left\langle \frac{1}{1 - q} \right\rangle, \quad \tau_2 = \left\langle \frac{q - 2qq_{\max} - q_{\max}^2}{(q - q_{\max})^2} \right\rangle. \tag{3.1}$$

Here $q_{\max} = \max\{s | F(s) = 0\}$ and $\langle \cdot \rangle$ indicates the averaging over the function $F(q)$. The time τ_1 is the time scale when the M th particle (the tagged particle) starts to move. The time τ_2 represents the time scale from which the configuration of the particles changes. Before τ_2 , the particles move in a single group whereas after τ_2 , the ‘‘platoon’’ structure appears. Note that in the left case in Fig.3, $\tau_2 = \infty$ and thus the particles keep going in a single group. On the other hand in the right case, we find that τ_2 is finite ($\tau_2 = 3$) and we see in this figure that during $\tau_1 M (\sim 693) < t < \tau_2 M (= 1200)$, they moves in a group while from $\tau_2 M < t$, they split into two group. There is a extremely slow particle in a group of the particles and at $t = \tau_2 M$ this particle begins to delay and makes the second group.

Now we discuss the dynamics of the M th particle during $\tau_1 M < t < \tau_2 M$. We scale $X(t_j)$ and t_j as

$$\tilde{x}(\tau_i, F) = \frac{X(t_i) - \tilde{A}(t_i/M, F)M}{\tilde{D}(u, F)M^{1/3}}, \tag{3.2}$$

$$t_i = uM + \tilde{C}(u, F)M^{2/3}\tau_i, \tag{3.3}$$

where \tilde{A} , \tilde{C} , and \tilde{D} are some constants which depend on u and F . Under the scaling, we have the following result:

$$\lim_{M \rightarrow \infty} \text{Prob}(\tilde{x}(\tau_1) \geq s_1, \tilde{x}(\tau_2) \geq s_2, \dots, \tilde{x}(\tau_m) \geq s_m | q_1, \dots, q_M) = \det(1 + \mathcal{K}_2 \mathcal{G}). \tag{3.4}$$

Here the right hand side represents the Airy process defined by Eqs. (2.14)-(2.16). In one time case ($m = 1$ in Eq. (3.4)), the corresponding result has been

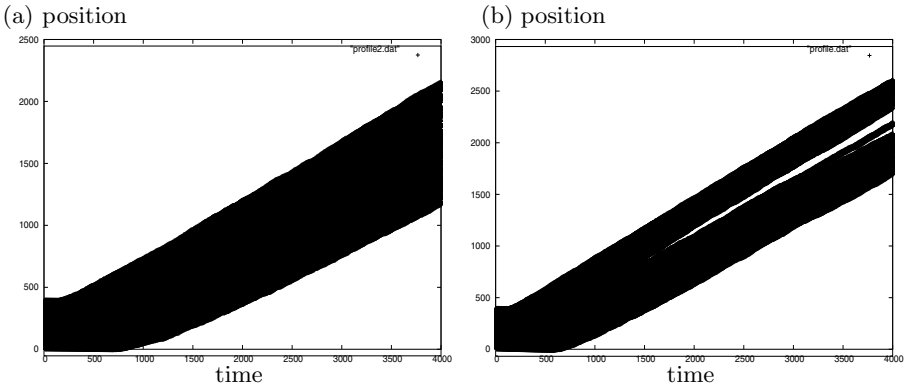


Fig. 3. Particle configurations in the TASEP with particle-wise disorder. ($M = 400$, $0 \leq t \leq 4000$) (a) $F(u) = 2 \times \mathbf{1}_{(0,1/2)}(q)$ (unit density). $\tau_1 = \log 2$ and $\tau_2 = \infty$ (b) $F(q) = 6(1 - 2q)^2 \mathbf{1}_{(0,1/2)}(q)$. $\tau_1 = 6 \log 2 - 3$ and $\tau_2 = 3$.

obtained by [15] in a disorderd growth model. The result (3.4) is the generation to the multi-time case.

We find that the Airy process also appears in the TASEP with particle-wise disorder. That is, the Airy process is universal in the sense that it is not destroyed by the disorder effect during $\tau_1 M \leq t \leq \tau_2 M$. However for $\tau_2 M \leq t$, the fluctuation property is expected to change. According to Ref. [15], we can expect that it is described by the simple Brownian motion. The verification of the conjecture would be an important topic in the future.

4 Concluding Remarks

In this paper, we have discussed the tagged particle property of the Totally Asymmetric Simple Exclusion process. Based on the technique in [7], we obtained the the multi-time distribution function of the position of the tagged particle in the situation where the hopping rates are generated independently by function F . We find that in the step initial condition (Fig. 1(b)), it is given by the Airy process, the largest eigenvalue process of a random Hermitian matrix (the GUE Dyson’s Brownian motion model).

It has been known that the distribution of a tagged particle position depends on initial configuration. In the alternating initial condition, where an occupied and empty sites are repeated alternately and in the case of a single hopping rate, the scaling limit is also described by the Fredholm determinant (2.14), but the kernel is not K_2 (2.18) but K_1 defined by

$$\begin{aligned} \mathcal{K}_1(\tau_1, \xi_1; \tau_2, \xi_2) &= -\frac{e^{-\frac{(\xi_2 - \xi_1)^2}{4(\tau_2 - \tau_1)}}}{\sqrt{4\pi(\tau_2 - \tau_1)}} \theta(\tau_2 - \tau_1) \\ &+ \text{Ai}(\xi_1 + \xi_2 + (\tau_2 - \tau_1)^2) e^{(\tau_2 - \tau_1)(\xi_1 + \xi_2) + 2(\tau_2 - \tau_1)^3/3}, \end{aligned} \tag{4.1}$$

where $\theta(x) = 1$ ($x > 0$) and 0 ($x \leq 0$) [16]. The stochastic process characterized by this Fredholm determinant is called the Airy_1 process [17]. In the single time case ($m = 1$), this is equivalent to the GOE Tracy-Widom distribution [18], the largest eigenvalue distribution of the random real symmetric matrix called the Gaussian Orthogonal Ensemble (GOE). It would be an interesting topic to study the fluctuation property of a tagged particle in more general initial configurations.

In the “two species” case where there are two types of particles with small and large hopping rates in the alternating initial condition, the tagged particle dynamics has been discussed recently [19]. The generalization to many-species cases would also be an interesting topic. How such kind of disorder affects the dynamical property of the TASEP is a challenging problem in the future.

References

1. Liggett, T.M.: *Interacting Particle Systems*. Springer, New York (1985)
2. Liggett, T.M.: *Stochastic Interacting Systems: Contact, Voter, and Exclusion Processes*. Springer, New York (1999)
3. Derrida, B., Evans, M.R., Hakim, V., Pasquier, V.: Exact solution of a 1D exclusion model using a matrix formulation. *J. Phys. A.* 26, 1493–1517 (1993)
4. Blythe, R.A., Evans, M.R.: Nonequilibrium steady states of matrix product form: a solver’s guide. *J. Phys. A.* 40, R333–R441 (2007)
5. Johansson, K.: Shape fluctuations and random matrices. *Commun. Math. Phys.* 209, 437–476 (2000)
6. Sasamoto, T.: Fluctuations of the one-dimensional asymmetric exclusion process using random matrix techniques. *J. Stat. Mech.*, P07007 (2007)
7. Imamura, T., Sasamoto, T.: Dynamics of a tagged particle in the asymmetric exclusion process with the step initial condition. *J. Stat. Phys.* 128, 799–846 (2007)
8. Kardar, M., Parisi, G., Zhang, Y.C.: Dynamic scaling of growing interfaces. *Phys. Rev. Lett.* 56, 889–892 (1986)
9. Okounkov, A., Reshetikhin, N.: Correlation function of Schur process with application to local geometry of a random 3-dimensional Young diagram. *J. Amer. Math. Soc.* 16, 581–603 (2003)
10. Borodin, A., Rains, E.M.: Eynard-Mehta theorem, Schur process, and their Pfaffian analogs. *J. Stat. Phys.* 121, 291–317 (2006)
11. Imamura, T., Sasamoto, T.: Correlation function of the Schur process with a fixed final partition. *J. Math. Phys.* 49, 053302 (2008)
12. Prähofer, M., Spohn, H.: Scale invariance of the PNG droplet and the Airy process. *J. Stat. Phys.* 108, 1071–1106 (2002)
13. Mehta, M.L.: *Random Matrices*, 3rd edn. Elsevier, Amsterdam (2004)
14. Tracy, C.A., Widom, H.: Level-spacing distributions and the Airy kernel. *Commun. Math. Phys.* 159, 151–174 (1994)
15. Gravner, J., Tracy, C.A., Widom, H.: A growth model in a random environment. *Ann. Prob.* 30, 1340–1369 (2002)
16. Borodin, A., Ferrari, P.L., Sasamoto, T.: Large time asymptotics of growth models on space-like paths II: PNG and parallel TASEP. *Commun. Math. Phys.* 283, 417–449 (2007)

17. Sasamoto, T.: Spatial correlations of the 1D KPZ surface on a flat substrate. *J. Phys. A* 38, L549–L556 (2005)
18. Tracy, C.A., Widom, H.: On orthogonal and symplectic matrix ensembles. *Commun. Math. Phys.* 177, 727–754 (1996)
19. Borodin, A., Ferrari, P.L., Sasamoto, T.: Two speed TASEP. *J. Stat. Phys.* 137, 936–977 (2009)

Chase and Escape in Groups

Atsushi Kamimura^{1,2}, Shigenori Matsumoto¹, Nobuyasu Ito¹,
and Toru Ohira^{1,3}

¹ Department of Applied Physics, The University of Tokyo,
7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

² Institute of Industrial Science, The University of Tokyo,
4-6-1, Komaba, Meguro-ku, Tokyo, 153-8505, Japan

³ Sony Computer Science Laboratories, Inc., 3-14-13, Higashi-gotanda, Shinagawa-ku,
Tokyo, 141-0022, Japan

Abstract. We study here a recently proposed theme of one group chasing another, called “group chase and escape”. Rather rich and complex behavior such as self-organized structures can arise from a model with simple rules. We discuss models with various cases of different speeds between the two groups, search ranges, and motion fluctuations.

Keywords: Chase and Escape, Pursuit and Evasion, Diffusion, Computer simulations.

1 Introduction

“Chase and Escape” or “Pursuit and Evasion” has been a topic of mathematics with a long history [1,2,3]. Original problems dealt with the case of one chaser pursuing a single target. Examples include a vessel chasing another, a lion chasing a prince with different conditions of speed ratios and boundaries. Rather complex trajectories have been found for these problems, often being challenging problems for obtaining analytical solutions.

With computational systems, this topic has been developing further. One such development is to consider problems of multiple chasers and/or evaders. Examples include predator-prey models that consider multiple entities chasing a single target or prey [4,5], multi-agent systems with applications to robotics [6,7], collective motions of self-driven particles [8,9] and Brownian particles with pursuit-and-escape interactions [10]. However, the problem of separate groups consists of large number of chasers and evaders has not been investigated.

Against this background, we have recently proposed a new paradigm of research problems called “group chase and escape” in which one group chasing another [11]. This research could find applications such as hunting of biological and computational viruses in real and cyber spaces [12,13], tracking of multiple vehicles or vessels. In this paper, we present some of the computational results of a simple model in this theme of group chase and escape.

2 Model

We describe our model in this section. The field is a two-dimensional square lattice $L_x \times L_y$ with periodic boundary conditions. Each site of the field is empty or occupied by one particle: a chaser or an evader (target).

The chasers and targets play tag by hopping between sites by chasing or getting away from the nearest opponent. Let us denote the positions of target and chaser as (x_T, y_T) and (x_C, y_C) , respectively. For each target, the distance to each chaser is calculated as $d = \sqrt{(x_T - x_C)^2 + (y_T - y_C)^2}$. Then a chaser with a minimum d is the nearest to the target. Here, if there are multiple nearest chasers, the target chooses one of them randomly. Then the target hops to its nearest site in the direction to increase the distance from this nearest chaser. The hopping rule is shown in Fig. 1. Generally, the target has two possible sites to which to hop, as shown in Fig. 1(a). In this case, one of the two sites is chosen with an equal probability $1/2$. If the $|x_T - x_C|(|y_T - y_C|)$ is zero, then the target has three possible sites to increase the distance, one of which is chosen with an equal probability $1/3$ (see Fig. 1(b)). If it happens that the chosen next hopping site is occupied by another target, it remains in its original site.

The moving rule for chasers is similar. They hop to get closer to their nearest targets. In the same manner as for the targets, we determine the nearest target for every chaser. The chaser hops to its nearest site to decrease the distance. Generally, the chasers choose one of their two possible sites to which to hop with an equal probability $1/2$. Here, if the $|x_T - x_C|(|y_T - y_C|)$ is zero, then the chaser hops only in $y-(x-)$ direction because hopping in $x-(y-)$ direction increases the distance. It does not move if the chosen next site is occupied by another chaser.

When a target is in the nearest site of a chaser, the chaser catches the target by hopping to the site, and then the target is removed from the system. After the catch, the chaser pursues the remaining targets in the same manner.

In accordance with the above hopping step, every chaser and target hops by one site. Here, we introduce hopping speeds of chasers and targets, denoted by V_C and V_T ($V_C, V_T \in \mathcal{N}$), respectively. We introduce discrete simulation steps and assume that, in each step, chasers and targets try to hop by V_C and V_T times, respectively. When $V_C = V_T = 1$, chasers and targets hop by one site in accordance with the above hopping rule. When the speed of chasers is faster, like when $V_C = r$ ($r > 1$) and $V_T = 1$, we first move both chasers and targets by one site, and after that we repeat the above procedure only for chasers by $(r - 1)$ times. When the speed of targets is faster, like when $V_T = s$ ($s > 1$) and $V_C = 1$, we repeat it for targets by $(s - 1)$ times after the one-site movement.

Initially, N_C^0 chasers and N_T^0 targets are randomly distributed over the lattice field. The number of targets, N_T , monotonically decreases along with the catch events, while the number of chasers, N_C , remains a constant N_C^0 . We run the simulations until all targets have been caught by chasers, i.e., $N_T = 0$, and the results are averaged over 10^4 runs.

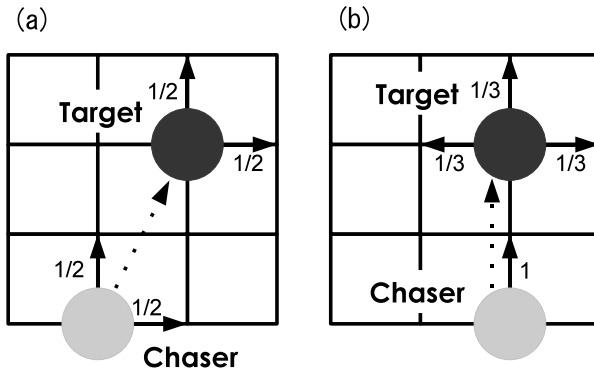


Fig. 1. The hopping rule for chasers(light grey circles) and targets(black circles). While chasers hop to get closer to their nearest targets, targets hop to get away from their nearest chasers. The dotted arrows from the chaser to the target indicate that the chaser hops to get closer to the target. The solid arrows show possible hopping directions with the indicated probabilities. (a) In the general case, they have two choices. (b) When the chasers or targets are in the same x or y -axis, chasers have one choice, while targets have three choices.

3 Simulation Results

3.1 Cost of Group Chase

Let the entire catch time T be the time it took the chasers to catch all the targets. Their distribution is shown in Fig 2(a). Since T can also be interpreted as a “lifetime” of the final target, Fig. 2(a) gives the probability distribution of its lifetime. When N_C is larger than N_T , we note this distribution basically shows a parabolic shape in the log-log scales, suggesting a lognormal distribution. This distribution is deduced even in a pure diffusion model (see appendix). However, as $N_C \approx N_T$, it deviates from the lognormal distribution, reflecting the effect of chase and escape.

The average length of time T decreases as the number of chasers increases. In the case where the speeds of chasers and targets are equal, $V_C = V_T$, an individual chaser can not catch up with targets, so it can not finish the job by itself. Instead, a group of chasers catches a target by surrounding it so that the target cannot escape from them. Although an individual chaser independently tries to catch a target, it seems as if the group of chasers cooperates to catch a target.

If we look at the lifetime distribution of all targets, then we obtain the results in Fig. 2(b). The distribution shows large drops at the left at first, then increases, and peaks at a typical time. After the peak, it decreases again. The first drop suggests that there is a large number of targets whose lifetime is one. This is because in the initial condition, targets can be positioned in the sites nearest to chasers, i.e., $d = 1$. Thus, the targets are caught by the chasers in the next

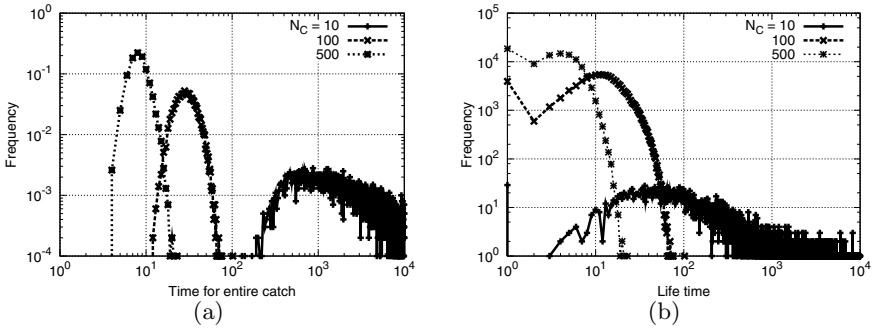


Fig. 2. (a) Distribution of the time for entire catch T , and (b) the lifetime distribution of targets for different N_C . The parameters for both plots are $L_x = L_y = 100$, $N_T^0 = 10$, $V_C = V_T = 1$.

step and its lifetime equals one. If the initial distances between targets and chasers are larger than one, the targets can momentarily get away from the chaser, causing the drop. After the drop, the frequency increases and we can see this distribution peaks, which can be inferred as a typical lifetime. This lifetime represents a timescale at which the group of chasers gathers around targets from the initial conditions and catches them. As the number of chasers increases, the timescale decreases. For comparison, we note the distribution in other cases. If we look at the distribution in the random walk model, we can see it exponentially decreases, so it does not drop or peak. If we examine a model in which chasers get closer to targets but targets follow the random walk processes, the peak appears to represent the timescale. However, the drop does not appear because the targets do not get away from chasers, so there is no drastic drop from the lifetime one to two.

In addition to the time for the entire catch T , we can introduce a “cost function” defined as $c = TN_C/N_T^0$. This quantity represents the unit cost for the group of chasers to finish the job per target. (The amount of work-hours N_C for which chasers are deployed (total cost) divided by the number of targets N_T .) We ask the question whether there is a “good” number of chasers N_C to deploy for a given number N_T^0 of targets.

In Fig. 3, we examine the cost by changing N_C for a fixed N_T^0 . We see that when the targets are as fast or faster than the chaser, there is a minimum in this unit cost. This means there is an optimal number of chasers N_C^* to finish the given group chase task most efficiently. When chasers are faster, however, we do not see such an optimal number of chasers. In this case, individual efficiency of the chasers are higher, a smaller number of them can finish the job in shorter time, and the cost monotonically rise along with the number of chasers.

When the targets are as fast or faster than the chasers, an individual chaser cannot catch up with targets, so it can not finish the job by itself. Instead, a group of chasers catch a target by surrounding it so that the target can not

escape from them. In this case, a number of chasers sufficiently larger than that of targets is necessary to finish the job efficiently. On the other hand, as the number of chasers exceeds the optimal number necessary to surround the targets, excessive chasers result in the increase of the cost.

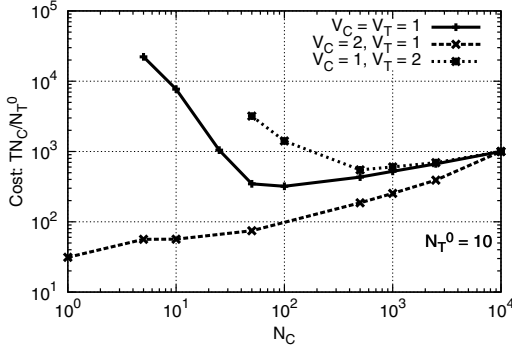


Fig. 3. The cost $c = TN_C/N_T^0$ vs. the number of chaser N_C for a fixed $N_T^0 = 10$ with different V_C and V_T

It is interesting to note that such a minimal cost is realized as a result of both of chase and escape processes. To illustrate this, we show the costs in different cases in Fig 4 both or either chaser and target follow a random walk process (see appendix). We see that when the targets are random walkers, the cost monotonically rises along with the number of chasers. On the other hand, when the chasers or both are random walkers, it monotonically decreases. The right side of the figure $N_C = 9990$ confirms that the system is fully occupied so that irrespective of their strategies, the targets are caught in one simulation step, leading to the cost $c = 1 * 9990/10 = 999 \sim 10^3$.

3.2 Issues of Range of Each Chaser

In our model, both chaser and targets have abilities to pick out the closest opponent. That means chasers can find targets over a limitless distance over the field. However, in reality, chasers search for targets in their vicinities. This is also the same for targets. Targets can recognize the existence of nearby chasers.

In order to reflect this factor, we introduce the search range l as follows. When a chaser searches for the nearest target, the search area is limited to the range $\sqrt{(x_T - x_C)^2 + (y_T - y_C)^2} < l$, where x_i, y_i denote the positions of targets($i = T$) and chasers($i = C$) in x and y -directions, respectively. If the chaser finds a target in the search range, it moves with the chase and escape hopping. If not, it follows the random-walk hopping (see appendix). The search range can be introduced in the same manner for the movement of targets.

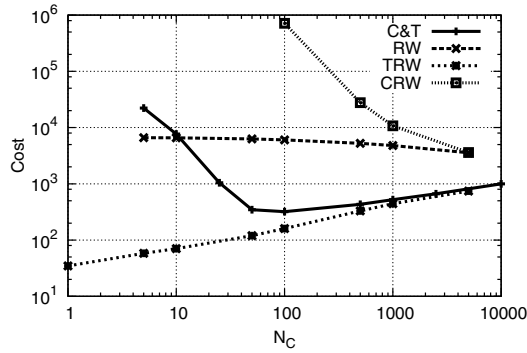


Fig. 4. The cost c vs. the number of chasers N_C for $N_T^0 = 10$ in the following four cases: original chasers and targets(C&T), both are random walkers(RW), Targets are random walkers(TRW) and Chasers are random walkers(CRW)

If the value of l equals zero, the movement is equivalent to the random walkers. On the other hand, the movement approaches the chase and escape hopping as the range increases to the system size.

Between these two cases, we have observed that the systems can show interesting behavior. One such example is shown in Fig. 5. Here, we set parameters so that there is no limit in the search range of targets, while the range of chasers is sufficiently short. For an appropriately low number of chasers, targets gather in relatively low-density areas of chasers and momentarily hide from chasers because the short-search-range chasers cannot recognize their existence. Only after a long time, chasers can find the group of targets and finally catch them. Such formations of spatial patterns and their relation to density fluctuations can be interesting issues.

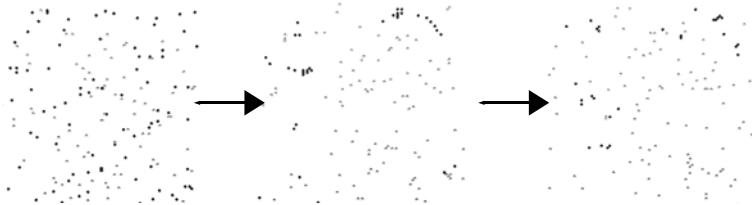


Fig. 5. The snapshots of the system with a time evolution from left to right. While targets(black circles) have unlimited search range, chasers(light grey circles) have the search range $l = 5$. The numbers of chasers and targets are fixed to $N_C = N_T^0 = 100$.

3.3 Issues of Long-Range Chaser Doping

We now further extend our model by introducing a non-homogeneity in the search ability among chasers. For example, in the group of chasers, some have a long search range, while the others follow the random walk hopping or have a short search range. We examine the cost c in such an example.

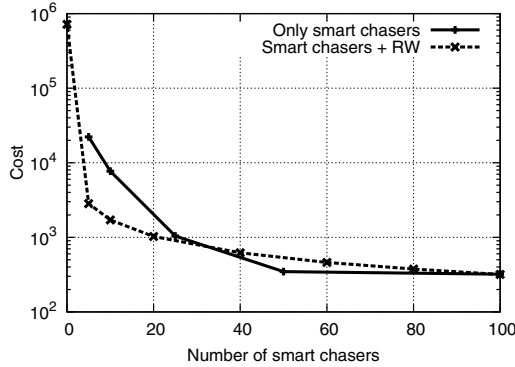


Fig. 6. The cost $c = TN_C/N_T^0$ by changing the number of smart chasers N_C^s for a fixed $N_T^0 = 10$. For the case of smart chasers and random walkers, we fix $N_C = 100$ so that $N_C - N_C^s$ random walkers also join tag. For the case of only smart chasers, $N_C = N_C^s$.

The group of chasers consists of two types: smart chasers and random walkers. The smart chasers have an unlimited search range, while the random walkers have search ranges of zero. In Fig. 6 we show the cost c by changing the number of smart chasers N_C^s with a fixed N_C . Hence, $N_C - N_C^s$ random-walking chasers also join the catch in the system. For comparison, we also show the case in which only N_C^s smart chasers are in the system and play tag. In both cases, we assume that targets have unlimited range. The left end 0 corresponds to the case in which all of the chasers are random walkers. The right end 100 corresponds to the case in which all chasers have unlimited ranges. As the ratio of smart chasers increases, the cost monotonically decreases. It is interesting to see that even a small number of smart chasers, say five to ten, can drastically drop the cost.

Comparison of the two cases is also of interest. If only a small number of smart chasers is available, which strategies will be better: let only the smart ones chase targets, or have random walkers join them? The group of random walkers also contributes to the catch events so we initially expect that the latter case is better. However, if we have to pay the salary per working hour to the chasers, the more chasers join, the more we have to pay. Our result in Fig. 6 indicates that for small number of smart chasers, the addition of random walkers helps to reduce the cost as it shortens the termination time T . However, as the number of smart chasers increases to 30 or 40, the former strategy with only smart chasers is better.

4 Discussion

In this paper, we have investigated a simple model under a theme of “group chase and escape”. By developing the cost function, we found characteristic behavior of group chase processes and evaluated efficiencies that have an optimum number of chasers. The values of the function were also compared among several cases, including the random walk. Our results confirm that the microscopic chase and escape rule has a scheme completely different from that of a reaction-diffusion system.

From another point of view, the problem of group chase and escape is an extension of studies of granular materials [14,15,16] to traffic problems [17,18], which have been actively investigated in recent decades. In the traffic problems, so-called “self-driven particles” are the basic constituents rather than physical particles like in granular materials. By giving them the aim of chase or escape, we are extending each unit further.

Even in biological molecule system, such as translation process of ribosomes [19], traffic jams are observed, and there have been studies of treating ribosomes as self-driven particles since 1969 [20]. We recently proposed a dynamical model in which ribosomes and the other related molecules to translation feel simple inter-molecule potentials and thermal noise. Still, we found they behave as if they are self-driven particles [21]. We may further consider how the nature of chase and escape emerges from interactions among biological molecules *in vivo*, e.g. neutrophil and bacterium [12].

In addition, we can include the effect of swarms [22,23]. The advantages and disadvantages of forming swarms are commonly studied in the fields of sociology, such as risk dilution. By comparing cases in which the targets/chasers are together or solitary, efficient survival/catch-up strategies could be developed. For example, instead of sensing the nearest target/chaser, each side could use “center of mass” of the locations, or more information about distributions of the opponent.

We may also include the effect of information transmission delay for chasers and targets to recognize the other particles’ positions. Delays often introduce unexpectedly complex effects into otherwise simple dynamical systems [24,25,26]. Some examples of applications are the modeling of blood cell reproduction [27], human posture and balance controls [28,29,30], traffic jams [31,32], and so on. Delays may induce interesting behavior in the context of chases and escapes.

Acknowledgement

The authors would like to thank Prof. John G. Milton of the Claremont Colleges for introducing us to the work of Nahin [3]. Toru Ohira thanks supports of the National Science Foundation of the United States of America (Grant No. 0617072).

References

1. Isaacs, R.: *Differential Games*. John Wiley & Sons, New York (1965)
2. Basar, T., Olsder, G.: *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics, Philadelphia (1999)
3. Nahin, P.J.: *Chases and Escapes: The mathematics of pursuit and evasion*. Princeton University Press, Princeton (2007)
4. Krapivsky, P.L., Redner, S.: Kinetics of a Diffusive Capture Process: Lamb Besieged by a Pride of Lions. *J. Phys. A: Math Gen.* 29, 5347 (1996)
5. Oshanin, G., Vasilyev, O., Krapivsky, P.L., Klafter, J.: Survival of an evasive prey. *Proc. Nat. Acad. Sci.* 106, 13696 (2009)
6. Hespanha, J.P., Kim, H.J., Sastry, S.: Multiple-Agent Probabilistic Pursuit-Evasion Games. In: *Proc. 38th Conference on Decision and Control*, p. 2432 (1999)
7. Vidal, R., Shakernia, O., Kim, J.H., Shim, D.H., Sastry, S.: Multiple-Agent Probabilistic Pursuit-Evasion Games with Unmanned Ground and Aerial Vehicles. *IEEE Trans. Robotics and Automation* 18, 662 (2002)
8. Vicsek, T., Czirok, A., Ben-Jacob, E., Cohen, I., Shochet, O.: Novel Type of Phase Transition in a System of Self-Driven Particles. *Phys. Rev. Lett.* 75, 1226 (1995)
9. Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. *Nature* 407, 487 (2000)
10. Romanczuk, P., Couzin, I.D., Schimansky-Geier, L.: Collective Motion due to Individual Escape and Pursuit Response. *Phys. Rev. Lett.* 102, 010602 (2009)
11. Kamimura, A., Ohira, T.: Group Chase and Escape. *New J. Phys.* 12, 053013 (2010)
12. Rogers, D.: *Crawling Neutrophil Chasing a Bacterium* (1950), <http://www.biochemweb.org/neutrophil.shtml>
13. Szor, P.: *The art of computer virus research and defense*. Addison Wesley Publishing Company, Upper Saddle River (2005)
14. Jaeger, H.M., Nagel, S.R., Behringer, R.P.: Granular solids, liquids, and gases. *Rev. Mod. Phys.* 68, 1259 (1996)
15. de Gennes, P.G.: Granular matter: a tentative view. *Rev. Mod. Phys.* 71, S374 (1999)
16. Kadanoff, L.P.: Built upon sand: Theoretical ideas inspired by granular flows. *Rev. Mod. Phys.* 71, 435 (1999)
17. Wolf, D.E., Schreckenberg, M., Bachem, A.: *Traffic and Granular Flow*. World Scientific, Singapore (1996)
18. Sugiyama, Y., Fukui, M., Kikuchi, M., Hasebe, K., Nakayama, A., Nishinari, K., Tadaki, S., Yukawa, S.: Traffic jams without bottlenecks - experimental evidence for the physical mechanism of the formation of a jam. *New J. Phys.* 10, 033001 (2008)
19. Alberts, B., Johnson, A., Walter, P., Lewis, J., Raff, M., Roberts, K.: *Molecular Biology of the Cell*, 5th edn. Garland Publishing, New York (2008)
20. MacDonald, C., Gibbs, J.: 1969 Concerning the kinetics of polypeptide synthesis on polyribosomes. *Biopolymers* 7, 707 (1969)
21. Matsumoto, S., Ito, N.: Dynamical Approach to Jam Phenomenon of Ribosomes. *The Mathematical Society of Traffic Flow* 15 (2009) (Japanese)
22. Bonabeau, E., Dorigo, M., Theraulaz, G.: *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Oxford (1999)
23. Couzin, I.D., Krause, J., Franks, N.R., Levin, S.A.: Effective leadership and decision - working in animal groups on the move. *Nature* 433, 513 (2005)

24. Glass, L., Mackey, M.C.: From Clocks to Chaos: The Rhythms of Life. Princeton University Press, Princeton (1988)
25. Ohira, T., Yamane, T.: Delayed Stochastic Systems. *Phys. Rev. E* 61, 1247 (2000)
26. Balachandran, B., Kalmar-Nagy, T., Gilsinn, D.E.: Delay Differential Equations: Recent Advances and New Directions. Springer, New York (2009)
27. Mackey, M.C., Glass, L.: Oscillation and Chaos in Physiological Control Systems. *Science* 197, 287 (1977)
28. Ohira, T., Milton, J.G.: Delayed Random walks. *Phys. Rev. E* 52, 3277 (1995)
29. Milton, J.G., Townsend, J.L., King, M.A., Ohira, T.: Balancing with positive feedback: the case for discontinuous control. *Phil. Trans. Roy. Soc. A* 367, 1181 (2009)
30. Milton, J.G., Cabrera, J.L., Ohira, T., Tajima, S., Tonosaki, Y., Eurich, C.W., Campbell, S.A.: The time delayed inverted pendulum: Implications for human balance control. *Chaos* 19, 026110 (2009)
31. Bando, M., Hasebe, K., Nakayama, A., Shibata, A., Sugiyama, Y.: Dynamical model of traffic congestion and numerical simulation. *Phys. Rev. E* 51, 1035 (1995)
32. Nakanishi, K., Itoh, K., Igarashi, Y., Bando, M.: Solvable optimal velocity models and asymptotic trajectory. *Phys. Rev. E* 55, 6519 (1997)

Appendix

In order to compare with the chase-and-escape hopping model, we consider random walk processes in which a particle hops to one of the four nearest sites with an equal probability $1/4$, irrespective of the positions of chasers and targets. Targets are caught when a chaser in the nearest site tries to hop to the site of the target.

When both chasers and targets follow the random walk processes, we call it the diffusion model. In this situation, dynamics of the number of targets N_T can be interpreted as a reaction-diffusion system in which targets are annihilated when they meet chasers, leading to a rate equation,

$$\frac{dN_T}{dt} = -kN_TN_C, \quad (1)$$

where k denotes a rate constant. As the number of chasers remains a constant, the solution gives $N_T(t) = N_T(0) \exp(-kN_C t)$. By rewriting the equation as $d \log(N_T)/dt = -kN_C$, we can also note the effect of fluctuations. If we assume that the rate constant k fluctuates with a normal distribution, the fluctuation of N_T will be given by a log-normal distribution.

A Velocity-Clearance Relation in the Rule-184 Cellular Automaton as a Model of Traffic Flow

Masahiro Kanai

Graduate School of Mathematical Sciences, The University of Tokyo,
Komaba 3-8-1, Meguro-ku, Tokyo 153-8914, Japan

Abstract. In this article, we investigate the velocity-clearance relation of vehicles in CA models for traffic flow using a calibration of the cell size (length) which we already proposed. We can mimic a more realistic behaviour of vehicles in traffic flow changing the cell size according to the density of particles in cellular automaton models. As a result, the velocity of particles in the rule-184 cellular-automaton model becomes dependent on the clearance, i.e., the distance to the next particle in front (in the right-hand side). Also, we show that the calibration is valid in that it reproduces a realistic flow-density diagram.

1 Introduction

Cellular automata (CA) have been used in physics [1,2,3] to model a wide variety of phenomena in a simple and intuitive manner. Also, there are actual discrete systems like CA, e.g., molecular motors on a microtubule [4]. In such systems, particles have a finite volume, and in principle, both collision and overtaking are forbidden. As a result, we often observe some particles stemming the flow by themselves. The above situation can be precisely mimicked with the so-called *exclusion rule* in CA models.

One of the most basic models [5,6,7,8,9,10,11,12] for traffic flow is the asymmetric simple exclusion process [13,14] (ASEP), a CA model incorporating randomness into an elementary CA rule 184 [1]. Particles in the ASEP hop to the next site with a given probability if it is not occupied. We should remark here that there are some possible choices to update cells [15,16], e.g., random sequential updating and shuffled dynamics [17], whereas all sites are synchronously updated in a CA [1]. However, we do not adhere to only the parallel update case in the following discussion.

The validation of traffic-flow models is carried out by comparing the simulation result to observational data [7,9,10,18,19,20,21]. Normal traffic data contains the density, velocity and flow, and one usually plots the velocity against the flow, the velocity against the density and the flow against the density. In particular, the significant properties of a traffic flow are summed up in the flow-density diagram [22]. Some simple models such as the ASEP are known to be exactly solvable [15,16,23,24], which means that one can obtain an exact flow-density diagram. Note that the flow-density diagram for the ASEP has the so-called

hole-particle symmetry: i.e., inverting all the cells in concert, one sees that the diagram is symmetric in the density. In the next section, we discuss a calibration of CA models for traffic flow focusing on these exactly solvable ones.

2 Calibration of the Cell Size in Cellular-Automaton Models

In this work, we simply consider ν vehicles of the same size c on a circuit of length l . The *real-world* density of vehicles, which is conserved in simulations, is accordingly defined as

$$\rho_{RW} = \frac{\nu c}{l} \quad (0 \leq \rho_{RW} \leq 1) . \tag{1}$$

Meanwhile, in the corresponding CA model the particle density is defined as

$$\rho_{CA} = \frac{N}{L} \quad (0 \leq \rho_{CA} \leq 1) , \tag{2}$$

where N is the number of occupied cells, and L is the total number of cells. Let d be the cell size. Since $\nu = N$ and $l = Ld$, we then have

$$\frac{\rho_{CA}}{\rho_{RW}} = \frac{d}{c} . \tag{3}$$

In this article, we shed light on the following condition with respect to vehicle size c and cell size d :

$$c \leq d \leq 2c . \tag{4}$$

This is justified because if $d < c$ then vehicles could not be contained in each single cell, or if $2c < d$ then one should reduce the cell size d to half. The CA models for traffic flow admit of optimizing the cell size according to the particle density. We therefore assume that the cell size d is determined depending on the density ρ_{RW} as $d = cf(\rho_{RW})$, where $f(\rho)$ is a function such that $1 \leq f(\rho) \leq 2$ due to (4). Accordingly, (3) yields

$$\rho_{CA} = \rho_{RW} f(\rho_{RW}) . \tag{5}$$

Note that since $0 \leq \rho_{CA} \leq 1$ by definition, we have to refine the condition on $f(\rho)$ to

$$1 \leq f(\rho) \leq \min\left\{2, \frac{1}{\rho}\right\} . \tag{6}$$

Function f should be chosen according to the target real-world system. In the next section, we show some elementary examples of the calibration applied for traffic-flow models.

3 Examples of the Calibration

As far as traffic-flow models are concerned, one may use a larger cell size while the density is low but a smaller one while it is high. Thus, we define $f(\rho)$ as a non-increasing function in ρ . For example, a simple choice, $f(\rho) = 2 - \rho$, leads to

$$\rho_{CA} = \rho_{RW}(2 - \rho_{RW}) . \tag{7}$$

In the following, we show the effects of this calibration applying (7) for basic traffic-flow models.

3.1 Asymmetric Simple Exclusion Process

As mentioned above, the ASEP is exactly solvable. The flow Q_{random} is described as a function of the density ρ_{CA} :

$$Q_{\text{random}}(\rho_{CA}) = p\rho_{CA}(1 - \rho_{CA}) , \tag{8}$$

where p is the hop probability. Note that (8) is for the random sequential updating. Then, using (7), the flow-density diagram, (8), is calibrated as

$$Q_{\text{random}}(\rho_{RW}) = p\rho_{RW}(2 - \rho_{RW})(1 - \rho_{RW})^2 . \tag{9}$$

(Note that we often denote different functions by the same symbol for the sake of simplicity.) In Fig. 1(a), we show the graphs of eqs. (8) and (9), finding that the calibration (7) helps to reproduce a more realistic flow-density diagram; in particular, the maximum-flow density moves from $1/2$ to $1 - 1/\sqrt{2}$ ($\simeq 0.29$).

Also, we consider the average velocity of vehicles as a function of the density. Following the assumption [22], $Q = \rho v$, with respect to flow Q , density ρ , and velocity v , we have

$$\begin{aligned} v_{\text{random}}(\rho_{CA}) &= p(1 - \rho_{CA}) , \\ v_{\text{random}}(\rho_{RW}) &= p(2 - \rho_{RW})(1 - \rho_{RW})^2 . \end{aligned} \tag{10}$$

Figure 1(b) shows the graphs of (10). We see density dependence of the mean hop probability in the ASEP. Note that since the cell size becomes larger than the vehicle size after the calibration, the average velocity is no longer equivalent to the mean hop probability and can take values larger than 1. Precisely to say, the cell length is multiplied by $f(\rho_{RW}) = 2 - \rho_{RW}$ as well as in (7).

In Fig. 1(c), we show the velocity-clearance diagram. We note here that both velocity and clearance mean their mean values in the steady state. The clearance of a vehicle is defined as the distance to the next vehicle ahead. Accordingly, the average clearance, denoted by h , is obtained from the density as

$$h = \frac{1 - \rho}{\rho} . \tag{11}$$

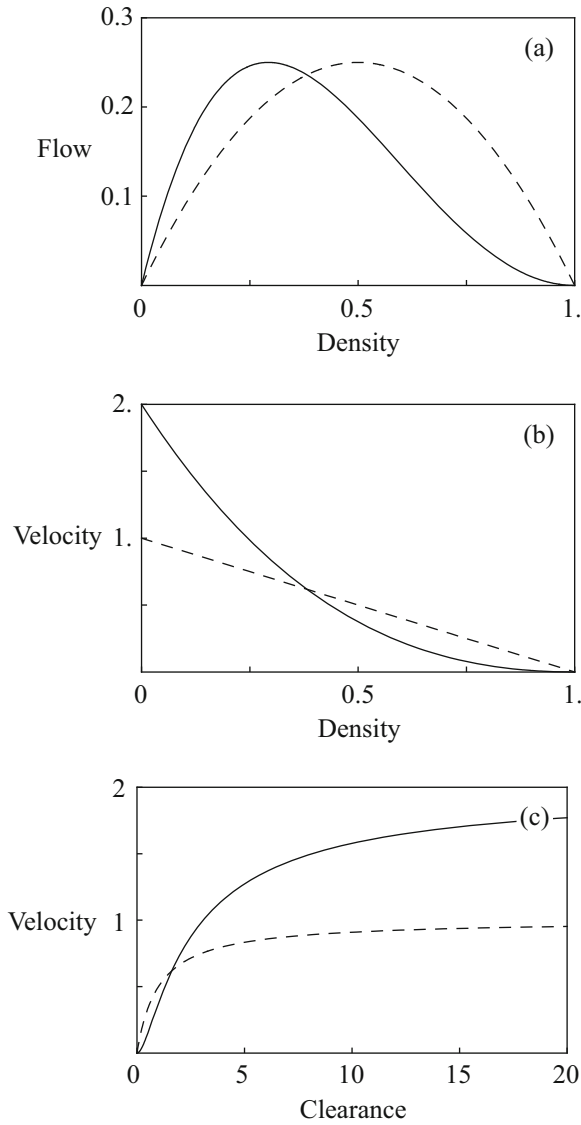


Fig. 1. The ASEP with random sequential updating and hop probability $p = 1$: (a) The flow-density diagram, (b) The velocity-density diagram and (c) The velocity-clearance diagram; (solid)calibrated and (dashed)non-calibrated

Using (10) and (11), we have

$$\begin{aligned}
 V_{\text{random}}(h_{CA}) &= p \frac{h_{CA}}{1 + h_{CA}} \quad , \\
 V_{\text{random}}(h_{RW}) &= p \frac{h_{RW}^2(1 + 2h_{RW})}{(1 + h_{RW})^3} \quad .
 \end{aligned}
 \tag{12}$$

Note that one may regard such a velocity-clearance relation as the so-called *optimal velocity function* introduced in the optimal velocity model [25] or in the Newell model [26].

3.2 Rule-184 Cellular Automaton

Next, we consider the rule-184 CA, and also the ASEP with the parallel updating for comparison. (Note that the ASEP with parallel updating includes the rule-184 CA as a special: $p = 1$.) The flow-density diagram for the parallel update case, which is described as [23]

$$Q_{\text{parallel}}(\rho_{\text{CA}}) = \frac{1}{2} [1 - \sqrt{1 - 4p\rho_{\text{CA}}(1 - \rho_{\text{CA}})}] \tag{13}$$

is calibrated, using (7), to be

$$Q_{\text{parallel}}(\rho_{\text{RW}}) = \frac{1 - \sqrt{1 - 4p\rho_{\text{RW}}(2 - \rho_{\text{RW}})(1 - \rho_{\text{RW}})^2}}{2} . \tag{14}$$

In Fig. 2(a), we show the graph of (14). The maximum-flow density calibrated is equal to $1 - 1/\sqrt{2}$, and is independent of p . Moreover, it is identical to that for the random sequential update case. In the special case of the rule-184 CA, we have

$$\begin{aligned} Q_{\text{parallel}}(\rho_{\text{CA}}) &= \min\{\rho_{\text{CA}}, 1 - \rho_{\text{CA}}\} , \\ Q_{\text{parallel}}(\rho_{\text{RW}}) &= \min\{\rho_{\text{RW}}(2 - \rho_{\text{RW}}), (1 - \rho_{\text{RW}})^2\} . \end{aligned} \tag{15}$$

In this case ($p = 1$ and the parallel updating), we see a first-order phase transition at the maximum-flow density, which distinguishes the model from the other ones. The phase transition implies that a traffic jam never occurs until the density exceeds the maximum-flow density. (By contrast, we find a traffic jam occurring at rather lower densities in the other cases.)

As well as in the previous case, the average velocity is obtained from the flow via $v = Q/\rho$. In particular, from (15), those for the rule-184 CA are given by

$$\begin{aligned} v_{\text{parallel}}(\rho_{\text{CA}}) &= \min\left\{1, \frac{1 - \rho_{\text{CA}}}{\rho_{\text{CA}}}\right\} , \\ v_{\text{parallel}}(\rho_{\text{RW}}) &= \min\left\{2 - \rho_{\text{RW}}, \frac{(1 - \rho_{\text{RW}})^2}{\rho_{\text{RW}}}\right\} . \end{aligned} \tag{16}$$

Figure 2(b) shows the graphs of (16), which presents how the proposed calibration works. Once the cell size is fixed, dynamics of the *real-world* vehicles is transformed into a coarse-grained motion in the CA model: Particles therein hop at most one site, with a constant probability and the exclusive interaction. The calibration then helps one restore the original dynamics. Compare Figs. 1 and 2 with the simulated/observed diagrams given in Refs. 7, 8, 21, 23, and 24 to see the effect of the present calibration.

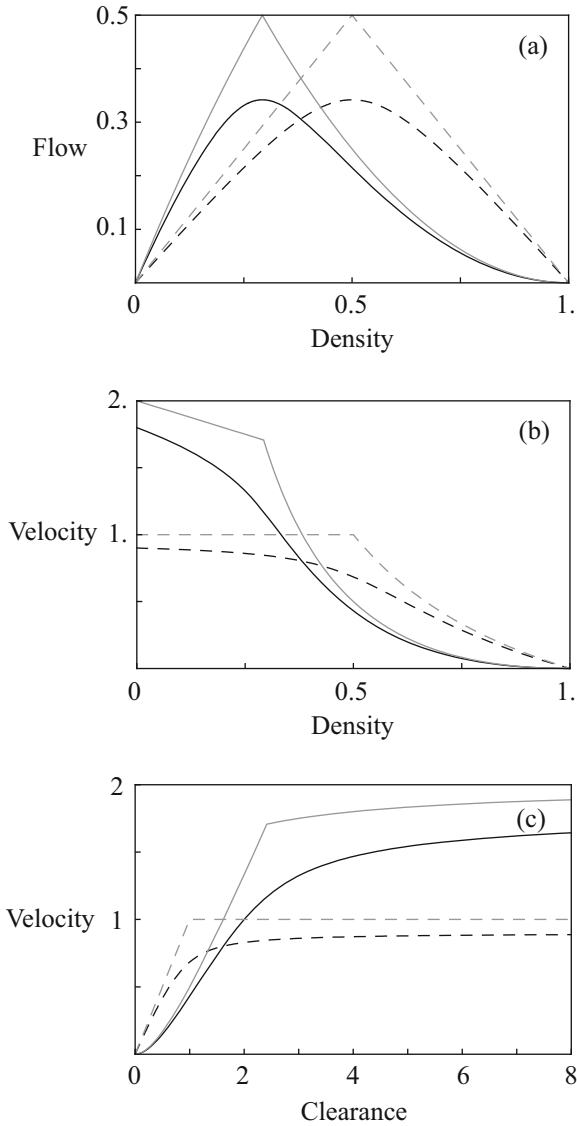


Fig. 2. The ASEP with the parallel updating and hop probability $p = 0.9$, and those of the rule-184 CA ($p = 1$): (a) The flow-density diagram, (b) The velocity-density diagram and (c) The velocity-clearance diagram. The curves therein are distinguished by the line style: (black)ASEP and (gray)Rule-184 CA; (solid)calibrated and (dashed)non-calibrated.

Figure 2(c) shows the velocity-clearance relation for the rule-184 CA obtained from (11) and (16) as

$$\begin{aligned}
 V_{\text{parallel}}(h_{CA}) &= \min\{1, h_{CA}\}, \\
 V_{\text{parallel}}(h_{RW}) &= \min\left\{\frac{1+2h_{RW}}{1+h_{RW}}, \frac{h_{RW}^2}{1+h_{RW}}\right\}.
 \end{aligned}
 \tag{17}$$

Note that the derivative of function $V_{\text{parallel}}(h_{RW})$ is discontinuous at $h_{RW} = 1 + \sqrt{2} (\simeq 2.4)$. This discontinuity is specific in the parallel-update dynamics; by contrast in the random-update dynamics, such a *phase transition* is lost. We think that the parallel update brings a kind of global correlation into the system and which plays a crucial role at the phase transition point. Among the researchers of traffic dynamics, the parallel updating is considered to be the best for CA modelling.

4 Summary and Remarks

In this article, we investigate the velocity-clearance relation of vehicles in CA models for traffic flow using the calibration of the cell size (length). Changing the cell size according to the density of particles in CA models, we can mimic more realistic behaviours of vehicles in traffic flow even if using one of the simplest CA models, i.e., the rule-184 CA. (For the avoidance of doubt, in the present work we does not intend a calibration-and-validation [27,28,29] of traffic-flow models.) In particular, the flow-density diagrams become realistic beyond the hole-particle symmetry, although further modification will be necessary to reproduce detailed structures such as a metastable branch observed in real-world traffic data.

Another motivation of ours is to reduce the cost of large-scale traffic simulations such as city traffic, using some simple model which reproduces a realistic traffic flow. In large-scale simulations, rapid computations are more important than realistic depictions. We remark that our central idea is that the cell size, which is usually set to the vehicle size, may also depend on other parameters (e.g., the number of vehicles conserved in each simulation), and it is not difficult to apply this idea for general CA models, such as two-dimensional CA models for pedestrian flow.

Finally, this article supplements a preceding work of ours [30] by a large extent with an investigation of the microscopic behaviour of vehicles by revealing a dependence of the velocity on the clearance. We will report in detail a close relation of the present calibration method to the optimal velocity model, as well as an extension for the open boundary conditions, in our subsequent publications.

The author is supported by Global COE Program, ‘‘The research and training center for new development in mathematics,’’ at Graduate School of Mathematical Sciences, The University of Tokyo.

References

1. Wolfram, S.: Cellular Automata and Complexity: Collected Papers. Addison-Wesley, Reading (1994)
2. DeMasi, A., Esposito, R., Lebowitz, J.L., Presutti, E.: Hydrodynamics of stochastic cellular automata. Commun. Math. Phys. 125, 127–145 (1989)

3. Lebowitz, J.L., Maes, C., Speer, E.R.: Statistical mechanics of probabilistic cellular automata. *J. Stat. Phys.* 59, 117–170 (1990)
4. Howard, J.: *Mechanics of Motor Proteins and the Cytoskeleton*. Sinauer Associates, Sunderland (2001)
5. Chowdhury, D., Santen, L., Schadschneider, A.: Statistical physics of vehicular traffic and some related system. *Phys. Rep.* 329, 199–329 (2000)
6. Helbing, D.: Traffic and related self-driven many-particle systems. *Rev. Mod. Phys.* 73, 1067–1141 (2001)
7. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. *J. Phys. I, France* 2, 2221–2229 (1992)
8. Fukui, M., Ishibashi, Y.: Traffic Flow in 1D Cellular Automaton Model Including Cars Moving with High Speed. *J. Phys. Soc. Jpn.* 65, 1868–1870 (1996)
9. Nagel, K., Wolf, D.E., Wagner, P., Simon, P.: Two-lane traffic rules for cellular automata: A systematic approach. *Phys. Rev. E* 58, 1425–1437 (1998)
10. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: A realistic two-lane traffic model for highway traffic. *J. Phys. A* 35, 3369–3388 (2002)
11. Kanai, M., Nishinari, K., Tokihiro, T.: Stochastic optimal velocity model and its long-lived metastability. *Phys. Rev. E* 72, 035102-5 (R)(2005); Kanai, M., Nishinari, K., Tokihiro, T.: Analytical study on the criticality of the stochastic optimal velocity model. *J. Phys. A* 39, 2921–2933 (2006)
12. Kanai, M., Isojima, S., Nishinari, K., Tokihiro, T.: Ultradiscrete optimal velocity model: A cellular-automaton model for traffic flow and linear instability of high-flux traffic. *Phys. Rev. E* 79, 056108 (2009)
13. MacDonald, C.T., Gibbs, H.: Concerning the kinetics of polypeptide synthesis on polyribosomes. *Biopolymers* 7, 707–725 (1969)
14. Spitzer, F.: Interaction of Markov processes. *Adv. Math.* 5, 246–290 (1970)
15. Rajewsky, N., Santen, L., Schadschneider, A., Schreckenberg, M.: The Asymmetric Exclusion Process: Comparison of Update Procedures. *J. Stat. Phys.* 92, 151–194 (1998)
16. Kanai, M., Nishinari, K., Tokihiro, T.: Exact solution and asymptotic behaviour of the asymmetric simple exclusion process on a ring. *J. Phys. A* 39, 9071–9079 (2006)
17. Wölki, M., Schadschneider, A., Schreckenberg, M.: Asymmetric exclusion processes with shuffled dynamics. *J. Phys. A* 39, 33–44 (2006)
18. Bando, M., Hasebe, K., Nakanishi, K., Nakayama, A., Shibata, A., Sugiyama, Y.: Phenomenological Study of Dynamical Model of Traffic Flow. *J. Phys. I, France* 5, 1389–1399 (1995)
19. Kerner, B.S., Rehborn, H.: Experimental Properties of Phase Transitions in Traffic Flow. *Phys. Rev. Lett.* 79, 4030–4033 (1997); Kerner, B.S.: Experimental Features of Self-Organization in Traffic Flow. *Phys. Rev. Lett.* 81, 3797–3800 (1998)
20. Tadaki, S., Nishinari, K., Kikuchi, M., Sugiyama, Y., Yukawa, S.: Observation of Congested Two-lane Traffic Caused by a Tunnel. *J. Phys. Soc. Jpn.* 71, 2326–2334 (2002)
21. Sugiyama, Y., Fukui, M., Kikuchi, M., Hasebe, K., Nakayama, A., Nishinari, K., Tadaki, S., Yukawa, S.: Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam. *New J. Phys.* 10, 033001 (2008); Nakayama, A., Fukui, M., Kikuchi, M., Hasebe, K., Nishinari, K., Sugiyama, Y., Tadaki, S., Yukawa, S.: Metastability in the formation of an experimental traffic jam. *New J. Phys.* 11, 083025 (2009)
22. Lighthill, M.J., Whitham, G.B.: On kinematic waves II. A theory of traffic flow on long crowded roads. *Proc. R. Soc. A* 229, 317–345 (1955)

23. Schadschneider, A., Schreckenberg, M.: Cellular automaton models and traffic flow. *J. Phys. A* 26, L679–L683 (1993)
24. Kanai, M.: Exact solution of the zero-range process: fundamental diagram of the corresponding exclusion process. *J. Phys. A* 40, 7127–7138 (2007)
25. Bando, M., Hasebe, K., Nakayama, A., Shibata, A., Sugiyama, Y.: Dynamical model of traffic congestion and numerical simulation. *Phys. Rev. E* 51, 1035–1042 (1995)
26. Newell, G.F.: Nonlinear Effects in the Dynamics of Car Following. *Oper. Res.* 9, 209–229 (1961)
27. Ossen, S., Hoogendoorn, S.P., Gorte, B.G.H.: Interdriver differences in car-following: a vehicle trajectory-based study. *Transp. Res. Rec.* 1965, 121–129 (2006)
28. Toledo, T., Koutsopoulos, H.N., Davol, A., Ben-Akiva, M.E., Burghout, W., Andreasson, I., Johansson, T., Lundin, C.: Calibration and validation of microscopic traffic simulation tools - Stockholm case study. *Transp. Res. Rec.* 1831, 65–75 (2003)
29. Brockfeld, E., Wagner, P.: Calibration and Validation of Microscopic Traffic Flow Models. In: Hoogendoorn, S.P., Luding, S., Bovy, P.H.L., Schreckenberg, M., Wolf, D.E. (eds.) *Traffic and Granular Flow 2003*, pp. 373–382. Springer, Heidelberg (2005)
30. Kanai, M.: Calibration of the Particle Density in Cellular-Automaton Models for Traffic Flow. arXiv:1002.1382, <http://arxiv.org/abs/1002.1382>

CA and MAS – With the NaSch as Example

Tobias Kretz

PTV Planung Transport Verkehr AG
Stumpfstraße 1, D-76131 Karlsruhe
Tobias.Kretz@ptv.de

Abstract. This is a contribution to the discussion of the interrelation between multi agent systems (MAS) and cellular automata (CA). The claim is that the Nagel Schreckenberg Model is a MAS, while at the same time it can perfectly be formulated as a CA. To underline this and to demonstrate the difference to and the benefit of a MAS formulation the rule table for the NaSch Model with $v_{max} = 2$ is given.

1 Introduction

The Nagel Schreckenberg Model [1] is one of the most prominent examples to describe a physical system using a cellular automaton. The property that it is a cellular automaton shows best with its deterministic limit ($p = 1$) for $v_{max} = 1$ being identical in the system's dynamics with CA-184 in Wolfram's notation of the 1-d, 3 cell neighborhood CA [2,3]. However, the Nagel Schreckenberg Model is only rarely formulated as a CA, it is always formulated in a driver's perspective with four action steps in each time step (accelerating, braking, dawdling, moving), a formulation, which here is argued to be a MAS. The CA formulation is – while it leaves the dynamics unchanged – very different, as can be seen in section 3 for $v_{max} = 2$.

2 Short Notes on Rule 184

Calling the left, central and right cell (l, c, r) and (b, w) as possible states black and white CA-184 (figure 1) can be formulated in logical form:

$$\text{if } (((l^t = b) \wedge (c^t = w)) \vee ((c^t = b) \wedge (r^t = b))) \\ \text{then } (c^{t+1} := b) \text{ else } (c^{t+1} := w)$$



Fig. 1. Rule table for CA-184 - The state of three cells determines the state of the center cell for the next round in the given way



Fig. 2. Compacted rule table for CA-184. Cells whose state does not matter (logical OR) are split diagonally.

This shows that the rule table for CA-184 can be formulated in a more compact way, as shown in figure 2. The compact representation immediately gives rise to two more interpretations of the rule, of which one is sufficient to determine the behavior of the system:

- For a black cell: “If there is no black cell to the right, move one cell to the right, otherwise stay.”
- For a white cell: “If there is no white cell to the left, move one cell to the left, otherwise stay.”

What happens in this interpretation step - without any change in the dynamics of the system - is that the cell-oriented view (the CA formulation) of a continuously existing static cell that can take one of two states changes to a particle-oriented view (in a sense a simple-MAS formulation) of a continued existence from one round to the other of either white or black particles. This of course is only possible for CA rules where the number of cells taking a certain state is conserved which is the case for CA-184.

3 Rule Table for the NaSch Model with $v_{max} = 2$

It is possible to formulate any deterministic v_{max} version of the Nagel-Schreckenberg model in just the same way as the original formulation of CA-184. The only thing one needs is a larger area of influence to the left and more colors to represent the different speeds. If one wants to distinguish between cars with $v_{max} - 1$ and v_{max} between the rounds (which is not necessary due to the acceleration step) the rule table contains $(v_{max} + 2)^{(v_{max} + 2)}$ elements. Already for $v_{max} = 2$ there are 256 elements. However, as shown in figure 2 quite a few reductions are possible. These lead to figure 3, which shows the compact rule table for the deterministic $v_{max} = 2$ version of the Nagel-Schreckenberg model.

The considerations so far show that even for $p = 0$ the CA formulation appears to be more complicated than the car-oriented formulation – not for a computer but for the human mind. This holds maybe even for $v_{max} = 1$ but in any case for $v_{max} = 2$ and other higher speeds. One also quickly realizes that the CA formulation becomes ever more complicated for increasing v_{max} as the number of possible elements in the rule table increases due to the increase of states and area of influence. For $v_{max} = 5$ one would have $7^7 = 823543$ elements. This could be reduced as it was done for $v_{max} = 2$ but the reduction process as well includes ever more computation steps. The car-oriented formulation however can be handled as easily for any v_{max} as it can be handled for $v_{max} = 2$.

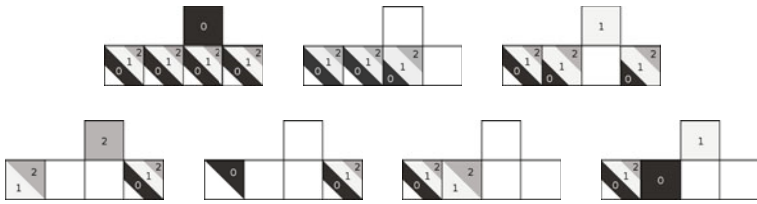


Fig. 3. CA formulation of the deterministic $v_{max} = 2$ version of the Nagel-Schreckenberg model. A white cell does not contain a car.

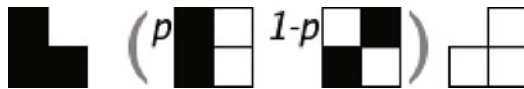


Fig. 4. CA formulation of the $v_{max} = 1$ version of the Nagel-Schreckenberg model. The brackets enclose alternatives of which one is chosen probabilistically. In cases where dawdling is possible, the state of two cells is fixed for the next round.

The last two steps are not necessary, but they reduce the number of elements. On the negative side they make the application of the rules appear more complicated. For $v_{max} = 2$ the result is shown in figure 5. In the context of complexity the crucial point is that the generation algorithm for the rule table becomes ever more complex in actual execution as well as it has an ever larger amount of input data (the deterministic rules) to be processed for increasing v_{max} .

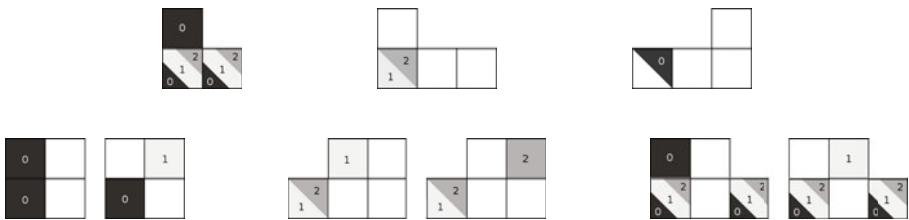


Fig. 5. CA formulation of the probabilistic $v_{max} = 2$ version of the Nagel-Schreckenberg model. Of the elements in the lower line, the left version (in the groups with two) is chosen with probability p and the right with $1 - p$.

It is suggested to view the (acceleration, braking, dawdling, moving) formulation of the model as a MAS formulation. In this formulation the probabilistic braking can be interpreted as reaction on a *belief* of the driver on what is going to happen, his *desire* is to reach his destination as soon as possible, and thus his *intention* to accelerate. Surely these elements of a MAS is modeled in a very basic way, but model complexity is not a good criterion to distinguish

MAS from non-MAS as “complexity” is a continuous quantity and it’s not clear where to make a cut. The evolution of models of pedestrian dynamics [4] over the years shows how it can happen that starting with clear CA models gradually one can lose central features and advantages of CA and develop models that would come closer to what clearly would be called MAS, while it still appears to be possible to re-formulate such models strictly as CA. See for example the model by Kaneda [5] which comes rather close to the cell-oriented formulation of the NaSch and is implemented in “artisoc”, an “agent based simulator” [6].

Clearly the CA formulation of the $v_{max} = 2$ Nagel Schreckenberg Model is far more complex than the $v_{max} = 1$ CA formulation. Already writing down the $v_{max} = 3$ CA formulation appears to be very arduous. In the MAS formulation, however, just a parameter changes from $v_{max} = 1$ to $v_{max} = 2$ to $v_{max} = 3$. Part of the reason for this difference might be that in a MAS formulation the number of agents is conserved manifestly, while it needs to be built into a CA formulation by carefully defining the rules.

While the cell-oriented (CA) and the particle-oriented (MAS) formulations appear to be quite different from each other, it appears difficult to find a clear criterion to distinguish MAS from non-MAS models. Since it is a discrete distinction, the clearest criterion appears to be, if a model is *formulated* particle-oriented (as distinguished from cell-oriented). As a consequence any particle-oriented microscopic model of a real multi agent system is a multi agent system itself and should be called such. This does not oppose the existence of very simple (low level) and very complex (high level) MAS.

References

1. Nagel, K., Schreckenberg, M.: A Cellular Automaton Model for Freeway Traffic. *Journal de Physique I* 2, 2221–2229 (1992)
2. Nagel, K.: Particle Hopping Models and Traffic Flow Theory. *Phys. Rev. E* 53, 4655–4672 (1996)
3. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)
4. Schadschneider, A., Klingsch, W., Klüpfel, H., Kretz, T., Rogsch, C., Seyfried, A.: Evacuation Dynamics: Empirical Results, Modeling and Applications. In: Meyers, R. (ed.) *Encyclopedia of Complexity and Systems Science*, vol. 5, pp. 31–42. Springer, Heidelberg (2009)
5. Kaneda, T.: Developing a Pedestrian Agent Model for Analyzing an Overpass accident. In: Waldau, N., Gattermann, P., Knoflacher, H., Schreckenberg, M. (eds.) *PED 2005*, Vienna, pp. 274–284. Springer, Heidelberg (2007)
6. Kozo Keikaku Engineering Inc. (KKE): artisoc. Simulation Software (2009)

Productivity Enhancement through Lot Size Optimization

Takumi Minemura¹, Katsuhiro Nishinari^{2,3}, and Andreas Schadschneider⁴

¹ School of Engineering, The University of Tokyo, Japan

² Research Center for Advanced Science and Technology, Univ. of Tokyo, Japan

³ PRESTO, Japan Science and Technology Agency, Japan

⁴ Institut für Theoretische Physik, Universität zu Köln

Abstract. We propose a simple model of a production plant that is based on the ASEP and simulate the dependence of the lead time T on the lot size r . Then, we derive an analytical description of the simulation results based on exact results for the single-species ASEP. Furthermore, we determine the optimum lot size r_c and investigate dependence of r_c on several parameters used in the simulations.

1 Introduction

For any production site, choosing the appropriate lot size is essential to improve production efficiency. Although methods for the determination of the optimum lot size have been proposed by management engineering [1] it is so far not possible to calculate this optimum for an actual production site.

In this study, we propose a simple model of a production plant that is based on the ASEP (asymmetric simple exclusion process) and is investigated by computer simulations and theoretical analysis. It allows to take into account the lot size and its influence on the productivity.

2 Definition of the Model

2.1 Modeling a Production Plant

A production plant is in general characterized by a complex network structure where many processes join and diverge each other. In order to simplify things, we first consider a one-dimensional model with a strictly linear structure (Fig. 1).

Let us assume that the plant produces N products which are divided into lots of r products [2]. In other words, the total production consists of $X = N/r$ sets where each set has r products. The production process contains n steps, corresponding to sites in the ASEP, in which the X sets move downstream the production line. Each step represents a process that has to be performed before

¹ In the following we will always assume that the lot size r is a positive divisor of N .

the product gets finished. We here assume that all processes require on average the same time. This is described as a stochastic process where each step forward occurs with the transition probability $p(r)$ if the next site of the production line is empty. Sets are inserted into the production line with the inflow probability $\alpha(r)$ and are removed at the end with the outflow probability $\beta(r)$.

Thus our model of a production line corresponds to a multispecies ASEP if each lot size r is identified with a different type of particle. Each species moves according to the rules of the ASEP with parallel dynamics, but the transition probabilities $p(r)$, $\alpha(r)$, $\beta(r)$ are in general different. In the following we will only the case of homogeneous lots, i.e. all lots have the same size r . Then the model reduces to the standard single-species ASEP with parallel dynamics.

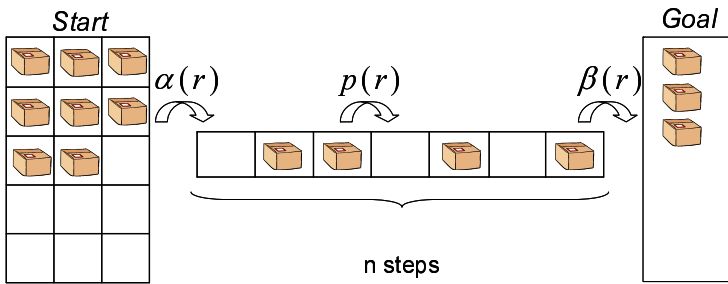


Fig. 1. Schematic representation of the production plant model

2.2 Transition Probabilities

Although the form of transition probability functions $p(r)$, $\alpha(r)$, $\beta(r)$ is in principle arbitrary, they should satisfy certain conditions if we want to apply the model to realistic situations.

The bulk transition probability $p(r)$ is related to the time $t(r)$ required at each step per set:

$$t(r) = \frac{1}{p(r)}. \tag{1}$$

In realistic situations, this time will become shorter, the smaller the lot number is, reflecting the advantage of making small lots. Mathematically this implies that $t(r)$ should be a monotonously increasing function, i.e. $p(r)$ is monotonically decreasing

$$\text{If } r_1 \leq r_2 \text{ then } p(r_1) \geq p(r_2). \tag{2}$$

On the other hand, there is also an advantage of making bigger lots since then the required time for each step per product becomes shorter. This can be expressed by the condition

$$\text{If } r_1 \leq r_2 \quad \text{then} \quad \frac{t(r_1)}{r_1} \geq \frac{t(r_2)}{r_2}, \tag{3}$$

or, equivalently,

$$r_1 p(r_1) \geq r_2 p(r_2). \tag{4}$$

A simple set of functions which satisfies the two conditions (2), (4) is given by the power law

$$p(r) = r^{-\gamma} \quad (0 < \gamma \leq 1). \tag{5}$$

When the exponent γ is small, the time required for each step per product decreases quickly for large r . On the other hand, when γ is close to 1, this decrease is rather small. Therefore, γ can be considered as an index of efficiency for the mass production. The efficiency becomes smaller for increasing γ .

3 Results from Computer Simulations

In the following we choose the transition probability function $p(r) = r^{-\gamma}$ and set $\alpha(r) = \beta(r) = p(r)$. Mainly we are interested in the lead time T required until all X sets have been finalized, i.e. reached the end of the production line. The parameters used in the simulations are summarized in Table 1.

Table 1. Parameters used in the simulations

Number of products	$N = 4320$
Number of steps	$n = 50$
Transition probability	$p(r) = r^{-\gamma}$
Inflow probability	$\alpha(r) = p(r)$
Outflow probability	$\beta(r) = p(r)$

Fig. 2 shows the lead times T obtained in the simulations as function of the lot size r in a double-logarithmic graph. Each points is obtained by averaging the result of 100 times simulations. The main results observed in the simulations are:

- When γ is small, T decreases monotonously.
- When γ is large, T increases monotonously.
- For intermediate values of γ , T has a local minimum.

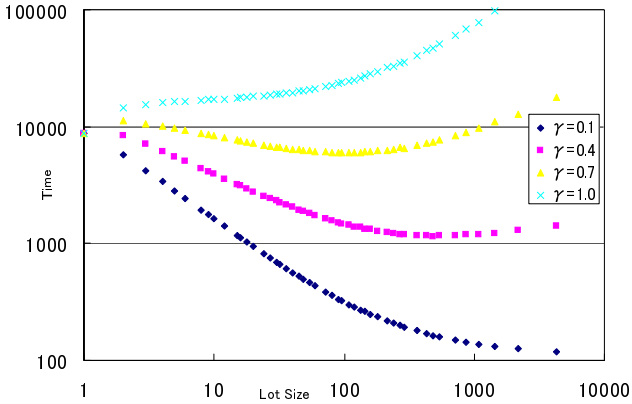


Fig. 2. Dependence of the lead time on the lot size

4 Theoretical Analysis

In the following we derive an analytical description of the simulation results based on exact results for the single-species ASEP.

In the simulation we have assumed that

$$p(r) = \alpha(r) = \beta(r). \tag{6}$$

For the single-species ASEP with parallel dynamics this corresponds to a point in the maximum current phase which is realized for $\alpha, \beta > 1 - \sqrt{1-p}$ [2,3,4]. Here, the current Q in the steady state is given by

$$Q = \frac{1 - \sqrt{1 - p(r)}}{2}. \tag{7}$$

The current Q expresses the number of sets that pass a certain point per unit time. Therefore, when all steps are in the steady state, the total time T_1 required before all X sets reach the end of the production line

$$T_1 = \frac{X}{Q} = \frac{2N}{r(1 - \sqrt{1 - p(r)})}. \tag{8}$$

Since the downstream process of the first set is always empty, the time T_2 required before the first set reaches the last (n th) step is,

$$T_2 = \frac{n}{p(r)}. \tag{9}$$

² For the case $\alpha = \beta = p$ studied here this is satisfied for all $p < 1$.

We now assume that all steps become stationary when the first set reaches the last step. Thus, the lead time T is

$$T = T_1 + T_2 = \frac{2N}{r(1 - \sqrt{1 - p(r)})} + \frac{n}{p(r)}. \tag{10}$$

Substituting $p(r) = r^{-\gamma}$ for $p(r)$ we have

$$T = \frac{2N}{r(1 - \sqrt{1 - r^{-\gamma}})} + nr^\gamma. \tag{11}$$

The results of this analytical theory are compared with simulations in Fig. 3. For all values of γ both are in excellent agreement.

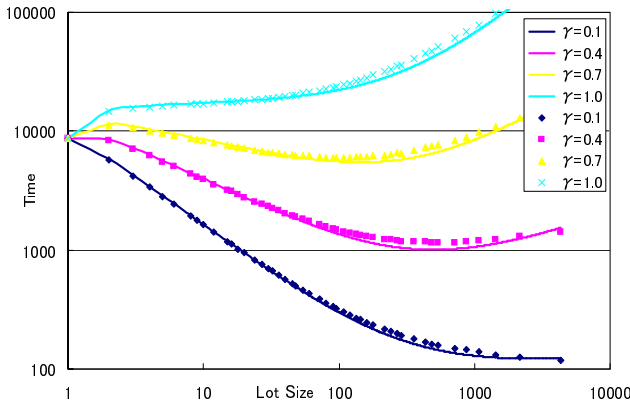


Fig. 3. Comparison of the simulation results (points) and the analytical theory (lines)

4.1 Other Transition Probability Functions

In derivation of the result (10) for the lead time T we have not used any specific assumptions about the transition probability function $p(r)$. Therefore, it is valid also for other choices than $p(r) = r^{-\gamma}$. Fig. 4 show results for the cases

$$p_1(r) = 1 - \left(\frac{r}{N}\right)^\gamma \quad \text{and} \quad p_2(r) = e^{-\gamma r}. \tag{12}$$

which also satisfy the criteria (2) and (4) We see that also for these choices of the transition probability function the agreement with the simulations is very good.

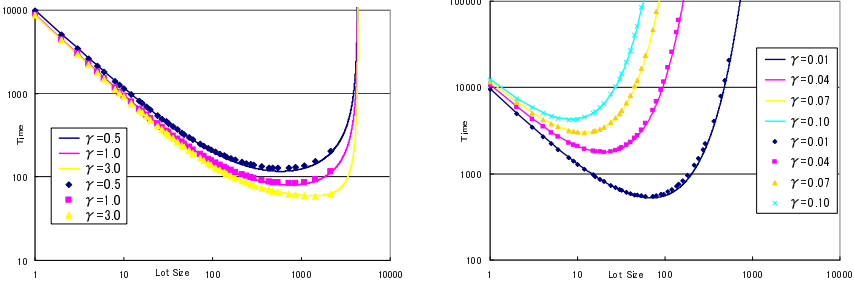


Fig. 4. Lead times as function of the lot size for the transition probability $p_1(r) = 1 - (\frac{r}{N})^\gamma$ (left) and $p_2(r) = e^{-\gamma r}$ (right)

4.2 Analysis of the Optimum Lot

We now determine the optimum lot size r_c which minimizes the lead time T from our theoretical solution. In the following we will use again the transition probability function $p(r) = r^{-\gamma}$. Differentiating (10) with respect to r gives

$$\frac{dT}{dr} = -\frac{2N}{r^2(1 - \sqrt{1 - p(r)})} - \frac{Np'(r)}{r(1 - \sqrt{1 - p(r)})^2 \sqrt{1 - p(r)}} - \frac{np'(r)}{p(r)^2}. \quad (13)$$

Substituting $p(r) = r^{-\gamma}$ for $p(r)$ then yields the following condition for the optimum lot size r_c :

$$\frac{dT}{dr} = -\frac{2N}{r_c^2(1 - \sqrt{1 - r_c^{-\gamma}})} + \frac{Nr_c^{-\gamma}\gamma}{r_c^2\sqrt{1 - r_c^{-\gamma}}(1 - \sqrt{1 - r_c^{-\gamma}})^2} + nr_c^{-1+\gamma}\gamma = 0. \quad (14)$$

This is a transcendental equation which has to be solved numerically for given parameters γ, N, n . Fig. 5 shows the dependence of r_c on the number N of products for a fixed number of steps n . Fig. 6 shows the dependence on n for

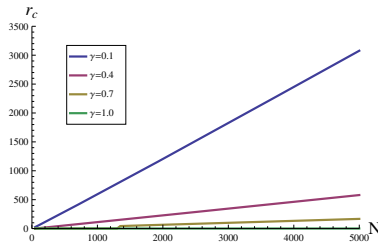


Fig. 5. Dependence of the optimum lot size on N for $n = 50$. For the case $\gamma = 1$, we have $r_c = 1$.

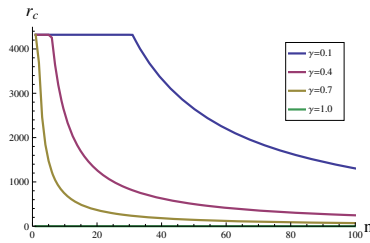


Fig. 6. Dependence of the optimum lot size on n for $N = 4320$. For the case $\gamma = 1$, we have $r_c = 1$.

a fixed number of products. For all values of γ , the optimum lot size increases linearly with N and decreases monotonously with n .

5 Conclusion

The lead time T can be approximated by the result (10). This result is valid also for other choices than $p(r) = r^{-\gamma}$. When $p(r) = r^{-\gamma}$, the optimum lot size increases linearly with N and decreases monotonously with n . In real operations, $\alpha(r)$, $\beta(r)$ and $p(r)$ may be defined by a deterministic rule. However since there are inevitable fluctuations in reality such as troubles of machines and delay in supply, we have treated them stochastically. Study on the other cases, such as $\alpha \neq \beta \neq p$, or extensions of this model to a realistic production network will be presented elsewhere.

Acknowledgement

This work is financially supported by the Economic and Social Research Institute, Japan.

References

1. Fujimoto, T.: Introduction of Production Management. Nikkei Publishing Inc. (2001)
2. Rajewsky, N., Santen, L., Schadschneider, A., Schreckenberg, M.: The Asymmetric Exclusion Process: Comparison of Update Procedures. *J. Stat. Phys.* 92, 151 (1998)
3. Evans, M.R., Rajewsky, N., Speer, E.R.: Exact solution of a cellular automaton for traffic. *J. Stat. Phys.* 95, 45 (1999)
4. de Gier, J., Nienhuis, B.: Exact stationary state for an asymmetric exclusion process with fully parallel dynamics. *Phys. Rev. E* 59, 4899 (1999)
5. Johansson, K.: Shape Fluctuations and Random Matrices. *Commun. Math. Phys.* 209, 437–476 (2000)

Multilane Single GCA-w Based Expressway Traffic Model

Anna T. Lawniczak^{1,2} and Bruno Di Stefano^{2,3}

¹ Department of Mathematics and Statistics, University of Guelph,
Guelph, Ontario N1G 2W1, Canada

² The Fields Institute for Research in Mathematical Sciences, 222 College Street,
Toronto, Ontario M5T 3J1, Canada

³ Nuptek Systems Ltd., Toronto, Ontario M5R 3M6, Canada

alawnicz@uoguelph.ca, bruno.distefano@nupteksystems.com

Abstract. We show how, by applying relaxation and extension of Elementary Cellular Automata Rule 184, we can model realistic highway traffic. In particular, we examine some of the available options for modeling variable acceleration, variable speed, multi-lane traffic, lane-changing for the purpose of avoiding obstacles or overtaking slower vehicles, suddenly stalled vehicles, and drivers' ability to account for the "brake lights" information and react to it. We describe the main features of our multilane expressway model developed adopting such extensions of Cellular Automata as "Global Cellular Automata" and "Global Cellular Automata with Write access".

Keywords: Cellular Automata, Global Cellular Automata, Global Cellular Automata with Write access, Highway/Expressway Traffic Modeling.

1 Introduction

Traditional highway traffic models of the 50s were macroscopic models considering vehicles like infinitesimally small particles in a fluid, i.e. the LWR model (Lighthill, Whitham, Richards), [1] and [2]. One problem of the LWR model and of models derived from it is a large number of parameters without an immediately intuitive equivalent when conducting empirical investigations. Microscopic traffic flow models were developed to solve this problem and explicitly describe vehicle interactions of the type that could be easily measured in the field, e.g. [3]. Road traffic is intrinsically discrete as each vehicle is one entity. The discrete entity point of view leads to consider Cellular Automata (CA) and Individual Based Modeling Methodologies (IBMM) allowing the development of "Virtual Laboratories", i.e. interactive environments for creating and conducting simulated experiments, capturing the emergence of large scale phenomena from discrete, local interactions and the effects of individual drivers' behaviour and local traffic fluctuations on the dynamics. Virtual laboratories can be used to study phenomena on various space and time scales, and test "what-if" and "how" hypothesis.

Elementary Cellular Automata (ECA) Rule 184 is often called “the traffic rule” and has been used for many road traffic models, e.g. see [4], [5], and [6]. Rule 184 is a rather simple traffic predictor, but it allows visualizing some of the well known patterns of real highway traffic, i.e.: platoons of moving cars separated by stretches of open road, when traffic is light, and “traffic waves” moving upstream, that is in opposite direction to the movement of the vehicles, i.e., waves of stop-and-go traffic, when traffic is heavy. ECA Rule 184 is good at modeling and visualizing, if adequate software is available, of qualitative properties of macroscopic, global, highway traffic phenomena. However, its usefulness to realistically model actual highway traffic at individual vehicle level is rather limited. In fact, ECA Rule 184, with periodic boundary conditions, can be used to model only unidirectional single-lane traffic, with no intersections and no entry or exit ramps, characterized by constant speed and null acceleration.

Even if one were to solve the problem of modeling variable speed, one would have to deal with other limitations such as the inability to distinguish a slow moving vehicle from a stalled vehicle or the inability to account for the “brake lights” information that actual drivers can see and react to. Dealing with multilane highway, vehicle passing, and different types of intersection (e.g., Yield-controlled, Stop-controlled, Signal-controlled, and Roundabout-based intersection) are even more complex problems.

In this paper we show how, by applying relaxation and extension of Rule 184, we can model more realistic highway traffic scenarios. Additionally, by extending the Cellular Automata (CA) paradigm to “Global Cellular Automata” (GCA) and to “Global Cellular Automata with Write access” (GCA-w) developed by Rolf Hoffmann and his collaborators we can mimic all deterministic qualitative behaviours of highway traffic, both at a global level and at individual vehicle level, [7], [8], [9], [10], [11], [12], and [13].

In this paper we do not present performance evaluation results which will soon be presented elsewhere. Also, we do not discuss modeling techniques in relations to entry & exit ramps, intersections, and road traffic in urban areas. Each subject requires much more space than available and each subject would justify one full paper. However, our model, presented in section 6, features entry & exit ramps and we briefly describe our implementation.

The scope of this paper is to present the architecture of our model and to explain how the adoption of the GCA and GCA-w paradigms can reduce the number of machine cycles to simulate the model and, thus, model longer expressways and longer periods of time.

2 CA & CA-Like Highway Traffic Modeling

Regardless how one represents an ECA, e.g. “truth table” or explicit Boolean expression, evolving a CA means computing equation (1) inside two loops, a space loop in which variable i varies from zero to the maximum size of the CA and a time loop in which variable t varies from zero to the maximum number of required discrete time steps. Equation (2) is the explicit form of equation (1) when the ECA rule under consideration is Rule 184.

$$c[t, i] := f[c[t-1, i-1], c[t-1, i], c[t-1, i+1]] \quad (1)$$

$$c_{i,t} = c_{i-1,t-1} \cap (\neg c_{i,t-1}) \cup (c_{i-1,t-1} \cup c_{i,t-1}) \cap c_{i+1,t-1},$$

which in C/C++ notation is: (2)

$$c_{i,t} = c_{i-1,t-1} \&\& (!c_{i,t-1}) \parallel (c_{i-1,t-1} \parallel c_{i,t-1}) \&\& c_{i+1,t-1}.$$

This architecture (i.e. this double loop) is extremely simple and powerful, but it requires being structurally unchanged whenever equation (1) is replaced by more complex rules of motion. Most practically applicable models are based on heuristics. As new information emerges while testing and validating the model, a large number of “If...Then” and “If ...Then ...Else” statements sneak stealthily into the code with many function calls. Consequently, the code becomes hard to verify for correctness and even harder to modify. Because of this, care must be taken to make sure that all situations to be accounted for can be expressed analytically as much as possible and that proper formalism of transmission rules and of data structures is maintained. A large number of models have been developed modeling realistic topologies, but as far as we know all of them have departed from the CA paradigm, e.g. [14], [15], [16], [17], [18], and [19].

Boccaro and Fuks have generalized Rule 184 to higher velocities, [20] and [21]. However, their work seems to be applicable only to periodic boundary conditions.

3 The Nagel – Schreckenberg Model

In 1992, Nagel and Schreckenberg proposed a stochastic model that probably established CA as a valid method of highway traffic, [22]. They extended the neighbourhood from one cell (as in ECA Rule 184) to five cells. They introduced six discrete velocities.

The model consists of four steps that have to be applied simultaneously to all cars:

- Acceleration
- Safety Distance Adjustment (“*slowing down due to other cars*”)
- Randomization
- Change of Position

During the “Acceleration” phase, at each time step, if the velocity of the vehicle at the end of the previous time step is $v < v_{\max}$, the velocity is incremented by one unit: $v \rightarrow v+1$. If the velocity of the vehicle at the end of the previous time step is $v = v_{\max}$, the velocity is left unchanged (null acceleration).

During the “Safety Distance Adjustment”, if a vehicle has d empty cells in front of it and its velocity v , after the Acceleration phase, would cause the vehicle to cover a distance larger than d , then the vehicle decelerates, that is, it reduces its velocity to d : $v \rightarrow \min\{d, v\}$. If the d cells in front of the vehicle are empty, no deceleration is required.

As stated by Nagel and Schreckenberg, the randomization “*is essential in simulating realistic traffic flow since otherwise the dynamics is completely deterministic. It takes into account natural velocity fluctuations due to human behaviour or due to varying external conditions.*” [22]. Randomization is an extension to the traditional deterministic paradigm of ECA and can be found in all realistic highway traffic models based on CA. The “Change of Position” assumes that the new velocity v_n for each car n causes advancing by v_n cells: $x_n \rightarrow x_n + v_n$.

It has been observed that the Nagel - Schreckenberg model is a “*minimal model*”, “*in the sense that any further simplification leads to a loss of realism*”, see [23].

The implementation of this model requires to modify the CA paradigm and to make the evolution of the CA not only dependent on the state of the neighbourhood but also on the current velocity of each vehicle. This implies that each cell is characterized not only by presence or absence of a vehicle but also by a pointer to a data structure containing the current velocity of the vehicle. Here we do not use the word “pointer” in the sense of the C/C++ programming language, but in the sense of “link, connection”.

Almost all models that we have examined implement variable velocity as in the Nagel - Schreckenberg model, the only substantial difference being the number of cells that the vehicle may need to advance to achieve its maximum speed.

Nagel and Schreckenberg write that “*Through the steps one to four very general properties of single lane traffic are modeled on the basis of integer valued probabilistic cellular automaton rules.*” We discuss multilane traffic modeling in next section.

4 Modeling Multilane Traffic

We know of:

- 2-D CA implementations
- Multi-CA implementations (i.e., one per lane)
- 1-D single GCA implementation

In the case of 2-D implementations the highway is represented by a CA consisting of a number of rows equal to the number of lanes being modeled and by a large number of cells representing the entire length of the highway. Lane changing is accomplished by simply moving to the adjacent cell on a different row. We developed a model of this type in the early stages of our research, [26].

Multi-CA implementations treat every CA as a separate road. The transition rules apply equally to every CA. Lane changing simply implies moving to the cell having the same cell number in the adjacent CA.

The 1-D single GCA implementation requires the extension of the CA model to a “Global Cellular Automata” (GCA), see [7], [8], [9], [10], and [11]. While a CA is characterized by the fact that the neighbourhood of each cell consists only of adjacent cells, the GCA is characterized by the fact that each cell has a neighbourhood consisting of all cells in the GCA.

In practical terms if one is modeling a highway long n cells and wide L lanes, we have the following situation:

- lane $l = 0$ spans from cell $c = 0$ to cell $c = n - 1$,
- lane $l = 1$ spans from cell $c = n$ to cell $c = 2 \times n - 1$
- lane $l = 2$ spans from cell $c = 2 \times n$ to cell $c = 3 \times n - 1$
-
- lane $l = L$ spans from cell $c = (L - 1) \times n$ to cell $c = L \times n - 1$

While this could be regarded as a matter of simple implementation, it has profound implications when implementing lane changing. In fact, while lane changing simply implies moving to the cell having the same cell number in the adjacent CA, when using a modeling the highway by means of a Multi-CA implementations (i.e., one CA per lane), in the case of 1-D single GCA implementation, if the vehicle is in cell c_i , any move to the next lane on the left is indeed a move to cell c_{i+n-l} any move to the next lane on the right is indeed a move to cell c_{i-n-l} , assuming that the lane $l = 0$ is the rightmost lane. Most compilers allow converting this operation to a single “*base plus index plus offset*” assembly language instruction, an extremely efficient, machine cycle parsimonious instruction.

The difference between a 2-D CA implementation and a 1-D single GCA implementation is that the first requires two space loops for each time step, while the second requires only one space loop. Even if the number of cells to be “visited” during each time step is identical, the one loop solution is more parsimonious in terms of machine cycles, because only one end-of-loop test must be performed instead of two loops prior to the increment of the loop counter.

5 Modeling Stalled Vehicles and Obstacles

One of the difficulties of modeling and simulating highway traffic by means of CA is that, at every time step, only the neighbourhood is known. The only information about past history is what can be derived from the current state of the neighbourhood. Thus, from the point of view of the vehicle moving at full speed, a fixed obstacle ahead (e.g., a stalled vehicle) is not distinguishable from a slow moving vehicle ahead. In real life, drivers do not have this problem because, at each time step, they:

- see further ahead than the location where their vehicle will move at the next time step
- can estimate the distance of the next vehicle ahead of them (sometime they can estimate even the distance of various vehicles ahead of them)
- remember where the vehicle was at the previous time step

The best way of modeling the driver’s reaction to a slow vehicle or to a stalled vehicle is to do exactly what a real driver does. If the leading vehicle moves at the maximum speed allowed, the distance between the two vehicles will remain unchanged. If the leading vehicle moves at a speed lower than the maximum speed allowed, after a few time steps the trailing vehicle will catch up and get very close, eventually to the point of having to slow down and, maybe, even stop. However, if we allow the trailing vehicle to look beyond its current position incremented by its maximum velocity, the trailing vehicle may estimate if the leading vehicle is moving slowly or if it is not

moving at all, i.e. it is stalled. In a multilane model, this information may be required to change lane and pass.

Modeling an unexpected obstacle other than a stalled vehicles, e.g. a fallen bridge or the load of a leading vehicle dropped on the road, introduces a new element in the CA or CA-like model. Traditional models assume that each cell is either empty or occupied by a vehicle. If we consider obstacles other than a stalled vehicle, we can have either an empty cell or a cell occupied by a vehicle or by an obstacle, i.e. a new species of particle.

One way of dealing with obstacles other than stalled vehicles, is to represent each cell with a class *Cell* consisting, among other things, of two private member Boolean variables, one indicating if the cell is occupied by a vehicle and the other indicating if the cell is occupied by an obstacle. If we call “B” (for “block”) private Boolean member variable *Obstacle* of cell *i*, formula (4) is the modified ECA Rule 184 capable of handling a hard, permanent, obstacle. In absence of any obstacle at cell *i*, $B_{i,t-1} = 0$ and $B_{i+1,t-1} = 0$, formula (4) is equivalent to (3), i.e. ECA Rule 184. If $B_{i,t-1} = 1$, $c_{i,t} = 0$, regardless of the value of $B_{i+1,t-1}$, any vehicle is prevented from reaching the cell occupied by the obstacle. If $B_{i,t-1} = 0$ and $B_{i+1,t-1} = 1$, cell *i* will be empty if no vehicle will reach it, otherwise it will be filled.

$$R_{i,t} = c_{i-1,t-1} \&\& (!c_{i,t-1}) \parallel (c_{i-1,t-1} \parallel c_{i,t-1}) \&\& c_{i+1,t-1}, \tag{3}$$

$$c_{i,t} = (R_{i,t} \parallel c_{i,t-1} \&\& B_{i+1,t-1}) \&\& (!B_{i,t-1}). \tag{4}$$

This is just a simple example, presented here for simplicity’s sake. In a similar and much more space consuming fashion, it is possible to derive and write formulas for handling obstacle when the model is capable of accounting for variable speed.

Modeling brake light has been implemented by various authors, see for instance [24] and [25]. In these models drivers react to the brake lights of the leading vehicle by braking too. The drivers observe only the brake lights of the nearest neighbour. For multilane models, the driver must decide if he/she should attempt to pass.

6 Our Model

We model the expressway as a number of adjacent lanes, where each lane is divided into cells. Each cell is assumed to be $m = 7.5$ in length as in most of the literature, e.g. [22] and [27]. This has been chosen because it corresponds to the space occupied by the typical car plus the distance to the preceding car in a situation of dense traffic jam. The traffic jam density is given by $1000/7.5m$ approximately equal to 133 vehicles per km.

Traffic is modeled applying the same algorithm at each time step, when each cell of each lane is examined in sequence and, if occupied by a vehicle, the vehicle navigation algorithm is applied. Thus, modeling traffic is equivalent to executing two large loops, an external time loop and an internal space loop. In reality, as we will see when describing the implementation the space loop is replaced by a number of loops where various operations are performed in sequence. Thus, after an initialization of all data structures used in the model, all execution time of the model is spent in these two loops.

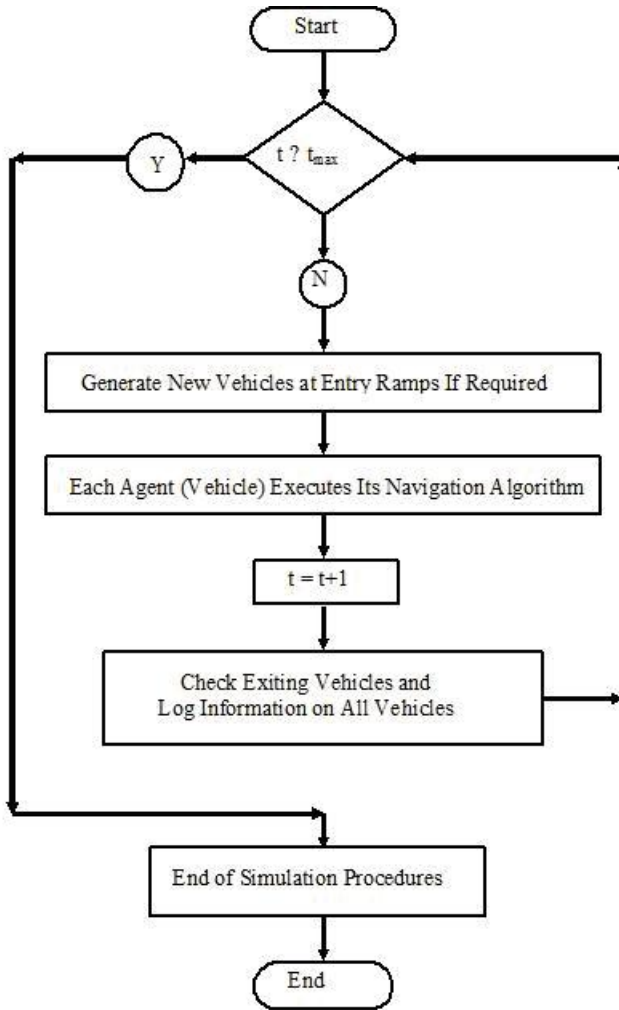


Fig. 1. Simplified flow chart showing the execution at each time step of the simulation

Figure 1 presents a simplified flow chart showing the execution at each time step of the simulation.

At each time step, vehicles are generated at each entry ramp according to a predefined vehicle generation probability that can be specified individually for each entry ramp. In the implementation, the software reads an input configuration file containing a description of the entire expressway. One of the predefined commands describes each entry ramps and the characteristics of the vehicles entering at that ramp. This command can be repeated multiple times for each entry ramp, indicating the different traffic characteristics at various times of the day (e.g., rush hour, day time, night time, work day, weekend, etc). For each instance of this command it is possible to specify:

- “Entry lane number” (always lane zero, that is the rightmost lane, except when entry cell is cell number zero, that is the entry to the expressway);
- “Entry cell number” (the location of the entry ramp from the beginning of the highway);
- “Start Time” and “End Time” measured in time steps from the beginning of the simulation when the specified creation probability applies;
- “Vehicle creation probability” during the specified time interval;
- Probability that the vehicle will be instantiated with the last cell of the expressway as its destination;
- “Maximum speed” that the vehicle will be able to reach while travelling on the expressway;
- Probability that the vehicle will be instantiated with a maximum speed equal to the one specified in the command.

The final destination probability and the maximum speed probability define not only the obvious probabilities implied by their names, but also the behaviour of the complementary probabilities. In other words if P_d is the probability that the vehicle is instantiated with last cell as its destination, $(1 - P_d)$ is the probability that the vehicle will go elsewhere, to other exit ramps. The specific exit ramp is assigned randomly. Similarly, if $P_{v_{max}}$ is the probability that the vehicle will be instantiated with the specified maximum speed, $(1 - P_{v_{max}})$ is the probability that the vehicle will be instantiated with a different maximum allowable speed. The specific different speed will be assigned randomly.

After all vehicles have been generated for each entry ramp, they are queued and placed on the ramp data structure (a first-in-first-out queue).

At this point, each vehicle on the highway, represented by a different instance of an agent, executes its navigation algorithm, that is the algorithm allowing changing lane, if required, advance, accelerate, decelerate, etc. The navigation algorithm is what we have described as a large conceptual space loop. Once the execution of this loop has been completed, time is incremented. We compare the predefined destination (exit ramp) of each vehicle with a neighbourhood of the cell where the vehicle is currently located. Those vehicles that have reached their exit ramp are removed from the expressway. Exit ramps are listed in the input configuration file without any other parameter than a keyword and the number corresponding to the cell where the exit ramp is located.

Information about all vehicles is logged to an output data file. This information is not aggregate information, but it is individual information about the location of each vehicle at the end of the execution of each time step. This output file allows calculating, off line, at the end of the simulation, the exact travel time of each vehicle, from entry ramp to exit ramp. The average of all the individual travel times is the travel time as earlier defined, see [28]. Aggregate information is output to a different data file where we store: current time step number, total number of vehicles instantiated, total number of vehicles on the road, and total number of vehicles delayed in entry ramps. Thus, it is possible to infer how many vehicles have exited at this time.

When the maximum simulation time, as defined in the input configuration file, is reached, some housekeeping work is carried on and the execution is terminated.

The navigation algorithm is divided into the following subsets:

- Change lane to the right if required (e.g., if no vehicle must be passed, as required by the rules of traffic, or if the vehicle is approaching its exit ramp);
- Change lane to the left if required (e.g., if a slower vehicle has to be passed or an obstacle has to be avoided);
- Advance, either at constant speed, if travelling at maximum (vehicle specific) speed, or accelerating/decelerating as it may be required by the traffic situation;
- Randomly, as specified by a command in the input configuration file, according to a predefined probability, execute an erratic behaviour if required.

Each subset of the navigation algorithm is executed as shown in Figure 2. For each of the above subsets, lane number and cell number are initialized to zero. Two buffers (i.e., arrays) are setup: OldBuffer is set up to contain a snapshot of the current traffic situation, with the location of each vehicle; NewBuffer is empty. For each lane, all cells are examined individually. If the cell is empty, i.e. there is no vehicle at that cell, nothing happens. If a vehicle is located in the cell under consideration, the algorithm required by the subset being executed is applied. All algorithms are of CA (Cellular Automata) like algorithms and are applied to the cell and a neighbourhood, i.e. a number of cells around it. Each lane is treated as a CA. Changing lane is logically equivalent to jumping from one CA to another one. However, the actual implementation uses 1-D single GCA, as previously described in Section 4. When all cells have been scanned and the related processing has been done, NewBuffer is copied into OldBuffer and the display is refreshed if the model is being executed in graphic mode.

Modeling multilane highway traffic with CA introduces some potential conflict whenever more than one vehicle “wants to move” to the same cell. This is not different from what happens in real life when, for example, a car is arriving at high speed on the leftmost lane and another car is changing lane from the centre lane to the leftmost lane. In real life, drivers can change their actions because time is continuous and because decision making is continuous and instantaneous. In a CA model, because all decisions are made based on the state of the CA at time $t-1$ and implemented at time t , we can have a conflict. In a traditional ECA, no vehicle can move ahead of another vehicle, so there is no conflict. In a 1-D CA, even if higher speeds are modelled, no vehicle can move ahead of another vehicle, so there is no conflict. In a 2-D CA, there is a potential conflict.

The “Global Cellular Automata with Write access” (GCA-w) developed by Rolf Hoffmann and his collaborators allows solving the potential conflicts, [12] and [13]. In the GCA-w model each occupied cell can have write access to the neighbours and can update its neighbours’ private member variable. Thus, before moving, a vehicle can issue a signal to the other vehicles in potential conflict and give them an early warning of its intention of moving to a given cell. This is simply done by setting a flag in a private member variable of the other vehicle.

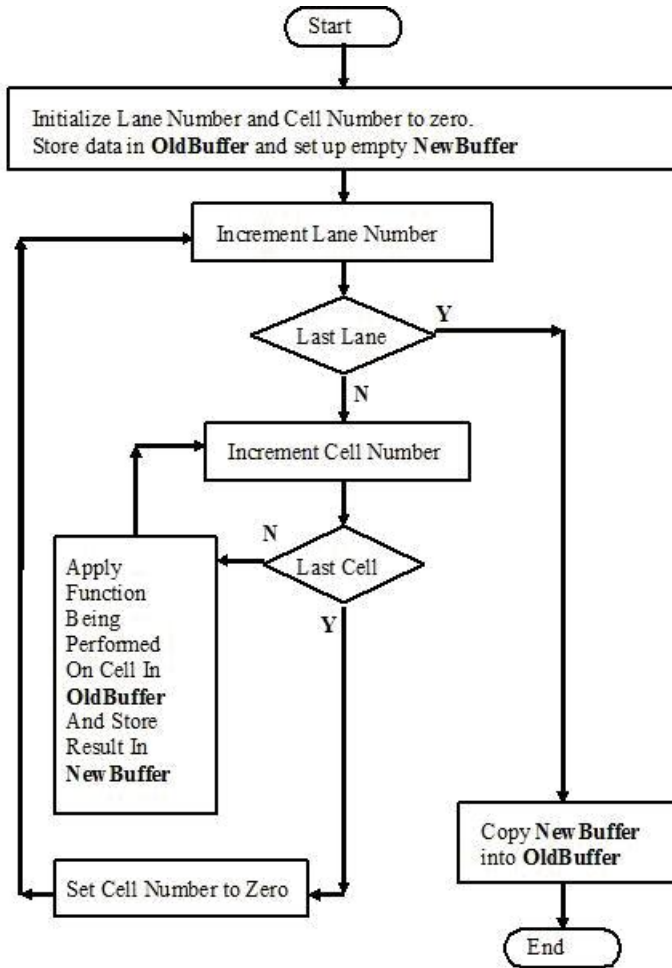


Fig. 2. Simplified flow chart showing the execution at each time step of the simulation

We have decided to assign the value of 3 seconds to each time step. Thus, the minimum speed of a vehicle advancing by one cell at each time step is equivalent to 9 km/h (that is, $7.5 \times 3600/3 = 7.5 \times 1200 = 9000$ m/h). This allows representing most realistic and legal speeds observed in Canadian expressways, with a vehicle advancing by a maximum of 11 cells per time step, that is, 99 km/h, as the speed limit is at 100 km/h. This is different from the model of Nagel and Schreckenberg, which uses 1 second per time step. We are currently comparing the results of our model with realistic traffic data in Ontario to decide if our choice is appropriate or needs to be revisited.

7 Software Implementation

We implemented our model using the C++ programming language. The execution can be in graphic mode and in non graphic mode. For graphic mode we used GLUT, the

OpenGL Utility Toolkit, [29]. We tested the graphic mode operation under MS Windows XP and MS Windows Vista. We tested the non graphic mode operation under MS Windows XP, MS Windows Vista, and Linux using SHARCNET, see [30]. Almost all storage data structures used in the code are vectors as defined the C++ Standard Template Library. All code written by us is compatible with *ISO/IEC 14882:1998*.

We used a pseudorandom number generator previously developed by us in the course of a different research project, [31] and [32]. We implemented the pseudorandom number generator as a class because the sequence of each instance used in the program must be independent of the others. “We are not content with one sequence of random numbers in the simulation system because we use them for different purposes”, see [32].

We called this implementation of our model *Freeway.exe*.

8 Future Work

We are currently validating our model with information from traffic engineers. We plan on using this model for practical traffic engineering applications, to estimate how some technological innovations affects travel time between two access ramps, an entry ramp and an exit ramp, once certain highway traffic parameters are known at certain points of the highway. Our concern is primarily with effects of flow and congestion through a long highway on travel time. The technological innovations include wireless communication from roadside transmitters to vehicles, wireless communication among vehicles, etc. While our model is intrinsically parallel, the current implementation is not parallel, but we are considering parallelizing our code for execution under SHARCNET, a consortium of Canadian academic institutions who share a network of high performance computers, see [30].

Acknowledgments. A. T. Lawniczak acknowledges partial financial support from the Natural Science and Engineering Research Council (NSERC) of Canada. B. N. Di Stefano acknowledges full financial support from Nuptek Systems Ltd. The authors thank Prof. Rolf Hoffmann for providing inspiring conversation. The authors acknowledge the use of Sharcnet computational resources and the hospitality of The Fields Institute for Research in Mathematical Sciences while writing this paper.

References

1. Lighthill, M.J., Whitham, G.B.: On kinematic waves: II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society A* 229, 317–345 (1955)
2. Richards, P.I.: Shockwaves on the highway. *Operations Research* 4, 42–51 (1956)
3. Cremer, M., Ludwig, J.: A fast simulation model for traffic flow on the basis of Boolean operations. *Mathematical and Computers in Simulation* 28, 297–303 (1986)
4. Fuk s, H.: Solution of the density classification problem with two similar cellular automata rules. *Physical Review E* 55(3), R2081–R2084 (1997)
5. Chopard, B., Droz, M.: *Cellular Automata Modelling of Physical Systems*. Cambridge University Press, Cambridge (1998)

6. Boccara, N.: *Modeling Complex Systems*. Springer, New York (2004)
7. Hoffmann, R., Volkman, K.-P., Waldschmidt, S.: Global Cellular Automata GCA: An Universal Extension of the CA Model. In: Worsch, T. (ed.) *ACRI Conference* (2000)
8. Hoffmann, R., Volkman, K.-P., Waldschmidt, S., Heenes, W.: GCA: Global Cellular Automata, A Flexible Parallel Model. In: Malyshkin, V.E. (ed.) *PaCT 2001*. LNCS, vol. 2127, p. 66. Springer, Heidelberg (2001)
9. Hoffmann, R., Volkman, K.-P., Heenes, W.: GCA: A massively parallel Model. In: *IPDPS 2003*. IEEE Comp. Soc, Los Alamitos (2003)
10. Ehr, C.: *Globaler Zellularautomat: Parallele Algorithmen*. Diplomarbeit, Technische Universität Darmstadt (2005),
<http://www.ra.informatik.tudarmstadt.de/forschung/publikationen/>
11. Heenes, W., Hoffmann, R., Jendrszczok, J.: A Multiprocessor Architecture for the Massively Parallel Model GCA. In: *IEEE Proceedings: 20th International Parallel & Distributed Processing Symposium, IPDPS/SMTPS 2006* (2006)
12. Hoffmann, R.: *Das massiv-parallele Berechnungsmodell GCA-w (Global Cellular Automata with Write-Access)*. Fachgebiet Rechnerarchitektur, Technische Universität Darmstadt, Internal Report (January 2009),
<http://www.ra.informatik.tu-darmstadt.de/forschung/publikationen/>
13. Hoffmann, R.: The GCA-w Massively Parallel Model. In: Malyshkin, V. (ed.) *PACT 2009*. LNCS, vol. 5698, pp. 194–206. Springer, Heidelberg (2009)
14. Chopard, B., Quelo, P.A., Luthi, P.O.: Traffic models of a 2d road network. In: *3rd European Connection Machine Users Meeting*, Parma, Italy (1995)
15. Chopard, B., Luthi, P.O., Quelo, P.A.: Cellular automata model of car traffic in two-dimensional street networks. *J. Phys. A* 29, 2325–2336 (1996)
16. Chopard, B., Dupuis, A., Luthi, P.O.: A Cellular Automata Model for Urban Traffic and its applications to the city of Geneva. In: *Proceedings of Traffic and Granular Flow* (1997)
17. Esser, J., Schreckenberg, M.: Microscopic simulation of urban traffic based on cellular automata. *Int. J. Mod. Phys. C* 8, 1025–1036 (1997)
18. Simon, P.M., Nagel, K.: Simplified cellular automaton model for city traffic. *Phys. Rev. E* 58, 1286–1295 (1998)
19. Dupuis, A., Chopard, B.: Parallel simulation of traffic in Geneva using cellular automata. In: Kühn, E. (ed.) *Virtual Shared Memory For Distributed Architectures*, pp. 89–107. Nova Science Publishers, Commack (2001)
20. Boccara, N., Fuk, H.: Cellular Automaton Rules Conserving the Number of Active Sites. *J. Phys A: Math. Gen.* 31, 6007–6018 (1998)
21. Fuk, H., Boccara, N.: Generalized deterministic traffic rules. *Journal of Modern Physics C9* (1), 1–12 (1998)
22. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. *J. Physique I* 2, 2221–2229 (1992)
23. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: Empirical test for cellular automaton models of traffic flow. *Phys. Rev. E* 70, 016115 (2004)
24. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: Towards a realistic microscopic description of highway traffic. *J. Phys. A* 33, L477 (2000)
25. Knospe, W., Santen, L., Schadschneider, A., Schreckenberg, M.: Single-vehicle data of highway traffic: Microscopic description of traffic phases. *Phys. Rev. E* 65, 056133 (2002)

26. Lawniczak, A.T., Di Stefano, B.N.: Development of CA model of highway traffic. In: Adamatzky, A., Alonso-Sanz, R., Lawniczak, A., Martinez, G., Morita, K., Worsch, T. (eds.) *Automata 2008: Theory and Applications of Cellular Automata*, pp. 527–541. Luni-ver Press (2008) ISBN-10: 1-905986-16-5, ISBN-13: 978-1-905986-16-3
27. Maerivoet, S., De Moor, B.: Cellular Automata Models of Road Traffic. *Physics Reports* 419(1), 1–64 (2005)
28. Transportation Engineering – Online Lab Manual, © 2000, 2001, 2003, Oregon State University, Portland State University, University of Idaho, Glossary (2000), http://www.webs1.uidaho.edu/niatt_labmanual/Chapters/TrafficFlowTheory/Glossary/index.htm
29. <http://www.opengl.org/resources/libraries/glut/>
30. <https://www.sharcnet.ca/>
31. Lawniczak, A.T., Gerisch, A., Di Stefano, B.: Development and Performance of Cellular Automaton Model of OSI Network Layer of Packet Switching Networks. In: *Proceedings of CCECE 2003*, Montreal Quebec, Canada, May 4-7 (2003)
32. Lawniczak, A.T., Gerisch, A., Maxie, K.P., Di Stefano, B.: *Netzwerk: Migration of a Packet-Switching Network Simulation Environment from MS Windows PC to Linux PC and to HPC*. In: *IEEE Proceedings of HPCS 2005: The New HPC Culture The 19th International Symposium on High Performance Computing Systems and Applications*, Guelph, May 15-18, p. 9 (2005)

Properties of Cellular Automaton Model for On-ramp System

Xin-Gang Li, Zi-You Gao, and Bin Jia

MOE Key Laboratory for Urban Transportation Complex Systems Theory and
Technology, Beijing Jiaotong University, Beijing 100044, P.R. China
lixingang@bjtu.edu.cn

Abstract. The on-ramp, as a typical bottleneck, has been widely studied by using Cellular Automata model. There are two different kinds of Cellular Automata models for on-ramp. This paper investigate the difference in traffic dynamics between the two kinds of models. The results show that they both have realistic and unrealistic features. The strong points of the two kinds of models should be combined together to model the on-ramp system.

1 Introduction

Nowadays, Cellular Automata (CA) model has become an excellent tool for simulating vehicular traffic flow [1,2]. It can reproduce most of the empirical features of traffic flow and has fast performance in computer simulation. In 1992, Nagel and Schreckenberg proposed the well-known NaSch model [3].

A traffic bottleneck is a section of road with a carrying capacity substantially below that characterizing other sections of the same road. It is the origin of traffic congestion. So that understanding the traffic dynamics around bottleneck is the key to solve traffic congestion. The on-ramp is a kind of representative bottlenecks. Many works have been done to investigate the traffic dynamics around the on-ramp [4,5,6,7,8]. There are two kinds of CA models for on-ramp: the acceleration lane model and the dummy ramp model. In the dummy ramp model, a ramp region is selected on the main road. Within each time step, the region is searched successively or stochastically for a vacant cell. Then a vehicle will be inserted into the cell with a probability. In the acceleration lane model, the ramp lane is also depicted by cells. The part of the ramp lane adjacent to the main road is called acceleration lane. The vehicle has to change from the acceleration lane to the main road.

In this paper, the properties of the two kinds of CA models for on-ramp system are analyzed and compared. Some common features of CA model for on-ramp are explored and the limitations of current model are given.

2 Model

In cellular automata traffic flow model, the road is divided into L cells, and a vehicle has a length of l cell(s). It is usually assumed that the length of a vehicle

is 7.5 m, then the length of a cell corresponds to 7.5/l m. So as l increasing, the space discretization is more finer. In the original NaSch model, $l = 1$ is selected. Here the refined NaSch model, in which l can be larger than 1, is used to describe the forward movement of vehicles. Next, we briefly recall the updating rules of the refined NaSch model. There are 4 sub-steps: (i) acceleration: $v_n(t + 1/3) \rightarrow \min(v_n(t) + 1, v_{\max})$; (ii) deceleration: $v_n(t + 2/3) \rightarrow \min(v_n(t + 1/3), d_n)$; (iii) randomization: $v_n(t + 1) \rightarrow \max(v_n(t + 2/3) - 1, 0)$ with probability p ; (iv) position update: $x_n(t + 1) \rightarrow x_n(t) + v_n(t + 1)$. Here $v_n(t)$ and $x_n(t)$ denote the velocity and position of the vehicle n respectively; v_{\max} is the maximum velocity; $d_n = x_{n+1}(t) - x_n(t) - l$ denotes the number of empty cells in front of the vehicle n ; p is the randomization probability.

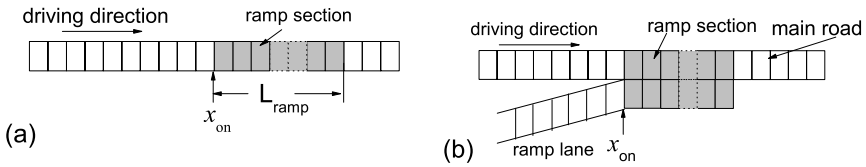


Fig. 1. Schematic illustrations of the on-ramp system. (a) the dummy ramp model; (b) the acceleration lane model.

The two common methods to model the on-ramp system are shown in Fig.1. In the dummy ramp model, the main road is represented by one lane, and the vehicle inserting region starts at the position x_{on} and has a length of $L_{ramp} \times l$ cells. The gaps of the vehicles in the region are calculated. Then the vehicle with the largest gap d_{\max} is signed and its position is denoted as x_{in} . If $d_{\max} \geq k_1$, which means the entrance gap is large enough for a vehicle, then a vehicle is inserted at the position $x_{in} + \lfloor d_{\max}/2 \rfloor$ with probability α_2 , and the velocity is set as the minimum value between the velocity of the leading vehicle and $\lfloor d_{\max}/2 \rfloor$. The function $\lfloor x \rfloor$ denotes the integer part of x .

In the acceleration lane model, besides the lane of the main road, there is also a ramp lane. The main road and the ramp lane joint together in the ramp section with the length of $L_{ramp} \times l$. In the ramp section, the vehicle on the right lane must change to the left lane before it reaches the end of ramp section, and the vehicle on the left lane is not allowed to change to the right lane. If the condition

$$d_{n,other} \geq \max(0, l + k_2) \text{ and } d_{n,back} \geq v_{ob} \tag{1}$$

is met, the lane-changing is performed by vehicle n . Here $d_{n,other}$, $d_{n,back}$ denote the number of free cells between the n th vehicle and its two neighbor vehicles on the destination lane at time t , respectively. If there is a vehicle on the destination lane drives side by side with vehicle n , $d_{n,back} = -l$. v_{ob} denotes the velocity of the following vehicle on the destination lane at time t . Condition $d_{n,other} \geq \max(0, l + k_2)$ means ‘‘I can move on the destination lane at next time step’’; and condition $d_{n,back} \geq v_{ob}$ is a safety criterion.

The open boundary conditions are adopted in the simulation. The inflow rates of the main road and the on-ramp lane are α_1 and α_2 respectively.

3 Simulation and Analysis

In this section, the simulation results are presented. In the simulations, the length of the main road is $L = 2000l$, and the ramp section starts at $x_{on} = 1000l$. The length of the ramp lane is $L/2$. The parameters $L_{ramp} = 5$, $v_{max} = 5l$ and $p = 0.3$ are used. One time step corresponds to 1 s. The flux on the left part of the main road is q_{left} and that on the right part of the main road is q_{right} . The flux on the ramp lane is q_{ramp} . In the dummy ramp model, q_{ramp} equals to the number of inserted vehicles per second. There is a relation that $q_{right} = q_{left} + q_{ramp}$. So q_{right} is the total flux of the ramp system.

The parameter l determines the acceleration rate of vehicle. As l increasing, the acceleration rate decreases. And the parameters k_1 and k_2 reflect the lane changing behavior of vehicle. As k_1 and k_2 increasing, the acceptance gap for lane changing becomes larger, that is to say, the lane changing behavior is more careful. The traffic dynamics influenced by those parameters have been investigated in our previous work [9,10]. Here we mainly focus on the difference in traffic dynamics between the two kinds of CA model.

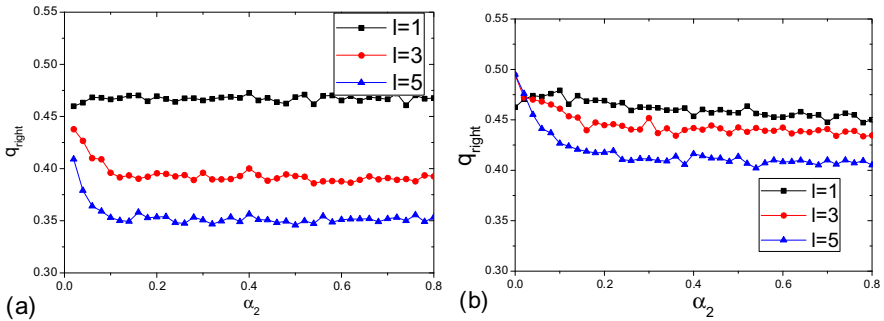


Fig. 2. The total flux as a function of α_2 with different l . (a) the acceleration lane model; (b) the dummy ramp model. The parameters $\alpha_1=0.5$, $k_1=2l$ and $k_2=0$ are used.

The total flux as a function of inflow rate of on-ramp are shown in Figs.2 and 3. Fig.2 shows the result with different l , and it reflects the influence of acceleration rate. One can see that as l increasing, the saturated flux decreases. Fig.3 shows the result with different k_1 and k_2 , and it reflects the influence of lane changing behavior. One can see that as k_1 and k_2 increasing, the saturated flux increases. The difference can be summarized as follows:

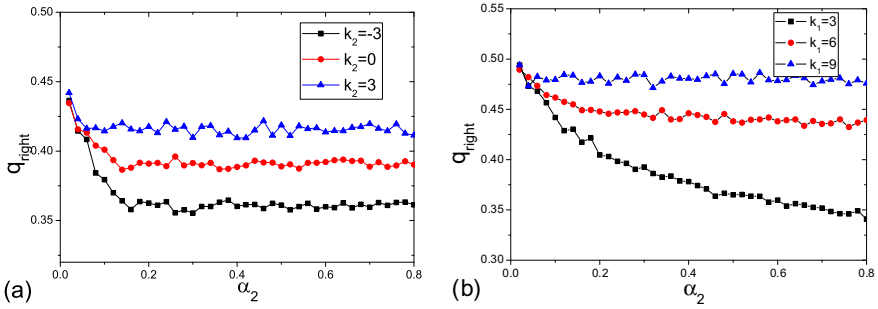


Fig. 3. The total flux as a function of α_2 with different k_1 and k_2 . (a) the acceleration lane model; (b) the dummy ramp model. The parameters $\alpha_1=0.5$ and $l = 3$ are used.

- 1) In the acceleration lane model, the total flux decreases to a saturated value as α_2 increasing. While in the dummy ramp model, the total flux decreases gradually as α_2 increasing. In the acceleration lane model, the vehicle changes into the main road according to the lane changing rule. When the traffic on the main road becomes saturated, the chance for lane changing is fixed. Then the flux on the ramp lane is also saturated. But in the dummy ramp model, the vehicle is inserted into the main road with certain probability when an acceptance gap has been found. The flux on the ramp lane is determined by the chance for finding an acceptance gap and the inserting probability. When the traffic on the main road is saturated, the chance for finding an acceptance gap is also fixed. Then the flux on the ramp lane is only determined by the inserting probability. It increases as α_2 increasing. Inversely, much disturbance is brought to the traffic on the main road. The total flux decreases gradually.
- 2) In general, the total flux of the dummy ramp model is higher than that of the acceleration lane model. In the dummy ramp model, the inserted vehicle is given an initial speed, which is equal to the speed of the former vehicle. Thus it can drive ahead with a speed and brings little disturbance to the traffic on the main road. In the acceleration lane model, the vehicle on the acceleration lane has to wait for a chance to change into the main road when the traffic on the main road is saturated. Thus it usually has a speed of 0 and needs to accelerate. This brings much disturbance to the traffic on the main road.
- 3) The acceleration lane model is more sensitive to the parameter l than the dummy ramp model. While the later is more sensitive to the acceptance gap. The disturbance to the traffic on the main road is determined by the acceleration time of lane changing vehicle from low speed to free speed. In the acceleration lane model, the lane changing vehicle usually has a speed of 0. The acceleration time is determined by the acceleration rate. While in the dummy ramp model, the lane changing vehicle is given a speed of the former

vehicle. Only when the acceptance gap is very small ($k_1 = l$), it could have a speed of 0. So the acceleration time in the dummy ramp model is mainly determined by the parameter k_1 .

In the acceleration lane model, most of the vehicles change lane with a speed of 0. This behavior is definitely unrealistic. The fact is that the vehicle on the ramp lane moves to the main road then it completes the lane change. So it has always gained a speed during the lane changing process. The acceleration lane model should reflect the realistic lane changing behavior. Considering that an initial speed is set to the inserted vehicle in the dummy lane model. We believe that the lane changing vehicle should also be set an initial speed when it changes into the main road in the acceleration lane model. This work will be done in our future work.

4 Conclusion

In this paper, simulations are carried out to study the traffic dynamics around an on-ramp. Different features of the acceleration lane model and the dummy ramp model are analyzed. In the acceleration lane model, the total flux q_{right} reaches a saturated value as α_2 increasing. While in the dummy ramp model, it decreases gradually as α_2 increasing. The acceleration lane model is more sensitive to the acceleration rate, while the dummy ramp model is more sensitive to the acceptance gap.

Acknowledgements

This work is partially supported by the National Basic Research Program of China No. 2006CB705500, the National Natural Science Foundation of China under Key Project No.70631001, Program for New Century Excellent Talents in University (NCET-07-0057), and the Fundamental Research Funds for the Central Universities (2009JBM049).

References

1. Chowdhury, D., Santen, L., Schadschneider, A.: Statistical Physics of Vehicular Traffic and Some Related System. Phys. Rep. 329, 199–329 (2000)
2. Maerivoet, S., Moor, B.D.: Cellular automata models of road traffic. Phys. Rep. 419, 1–64 (2005)
3. Nagel, K., Schreckenberg, M.: A Cellular automaton model for freeway traffic. J. Phys. I 2, 2221–2229 (1992)
4. Campari, E.G., Levi, G.: A cellular automata model for highway traffic. Eur. Phys. J. B 17, 159–166 (2000)
5. Diedrich, G., Santen, L., Schadschneider, A., Zittartz, J.: Effects of on- and off-ramps in cellular automata models for traffic flow. Int. J. Mod. Phys. C 11, 335–345 (2000)

6. Pederson, M.M., Ruhoff, P.T.: Entry ramps in the Nagel-Schreckenberg model. *Phys. Rev. E* 65, 056705 (2002)
7. Jia, B., Jiang, R., Wu, Q.S.: The effects of accelerating lane in the on-ramp system. *Physica A* 345, 218–226 (2005)
8. Jiang, R., Jia, B., Wu, Q.S.: The stochastic randomization effect in the on-ramp system: single lane main road and two lane main road situations. *J. Phys. A* 36, 11713–11723 (2003)
9. Li, X.G., Gao, Z.Y., Jia, B.: Cell size: an important factor for modelling bottleneck traffic in cellular automata model. In: *Proceedings of the Second International Joint Conference on Computational Science and Optimization, CSO 2009, Sanya, China, April 24-26, vol. 2, pp. 72–76 (2009)*
10. Li, X.G., Gao, Z.Y., Jia, B., Jiang, R.: Traffic behaviors of on-ramp system with cellular automata model. *Chin. Phys. B* (in press)

Inversion of Flux between Zipper and Non-Zipper Merging in Highway Traffic

Ryosuke Nishi^{1,2}, Hiroshi Miki³, Akiyasu Tomoeda⁴,
Daichi Yanagisawa^{1,2}, and Katsuhiko Nishinari^{5,6}

¹ School of Engineering, The University of Tokyo, Japan

² Research Fellow of the Japan Society for the Promotion of Science, Japan

³ Department of Material Sciences, Interdisciplinary Graduate School of
Engineering Sciences, Kyushu University, Japan

⁴ Meiji Institute for Advanced Study of Mathematical Sciences,
Meiji University, Japan

⁵ Research Center for Advanced Science and Technology,
The University of Tokyo, Japan

⁶ PRESTO, Japan Science and Technology Agency
tt097086@mail.ecc.u-tokyo.ac.jp

Abstract. For easing heavy traffic jams on weaving and merging sections on highway traffic, we compare the efficiency of zipper merging and non-zipper merging. Zipper merging is the merging of vehicles on two lanes by turns and achieved by only vehicle-to-vehicle interactions before merging. Non-zipper merging is the merging without any interactions before merging. In this comparison we use a cellular automaton model on multiple lanes with slow-to-start rules. Simulations and mean-field analysis show that the flux of zipper merging is larger (smaller) than that of random merging in the case of large (small) slow-to-start effect.

Keywords: Traffic flow, Zipper merging, Slow-to-Start Rule.

1 Introduction

In recent years, traffic dynamics has attracted much interest of mathematicians and physicists, so that it has been studied more and more diligently [1,2,3]. Researchers have mainly analyzed the traffic flow on one-lane roads by using car-following models [4,5] and cellular automaton (CA) models [6,7,8,9]. These days, vehicular traffic on multiple-lane roads with an intersection or a junction is expected to be analyzed in order to ease traffic jams and has been studied by using game theory [10,11], agent based simulations [12], and the simulations which includes the effect of cooperation between vehicles on two lanes [13].

However, these previous works did not study in detail the configuration of vehicles before merging, which is crucial to the efficiency of merging. Among various configurations, our previous work [14] discussed the alternative configuration because it realizes “zipper” merging, which is the merging of vehicles on two lanes alternatively. To induce it, we proposed a simple and bottom-up

method, which is to draw a compartment line between two lanes on the area inside merging sections as shown in Fig. 1 (a-b). This line prohibits vehicles from changing lanes, while permits them to see other vehicles on another lane. Vehicles moving along this line are expected to see other vehicles on another lane and to adjust their own configuration to alternative one. Our method is incredibly inexpensive than other methods for easing traffic jams, e.g., constructing cubic merging. Moreover, our method does not demand all vehicles to attach wireless communication instruments [15,16]. In our previous work we focused on the transformation of configurations before merging, and showed that alternative configurations are emergently achieved only by repulsive vehicle-to-vehicle interactions, by using CA simulations and mean-field calculations.

For further study, in this paper we discuss the effect of vehicle-to-vehicle interactions on the flux of merging. Thus, we compare the flux with vehicle-to-vehicle interactions before merging and that without any interactions by numerical simulations and mean-field calculations. In this investigation we use a multiple-lane CA model with slow-to-start rule [17,18,19,20].

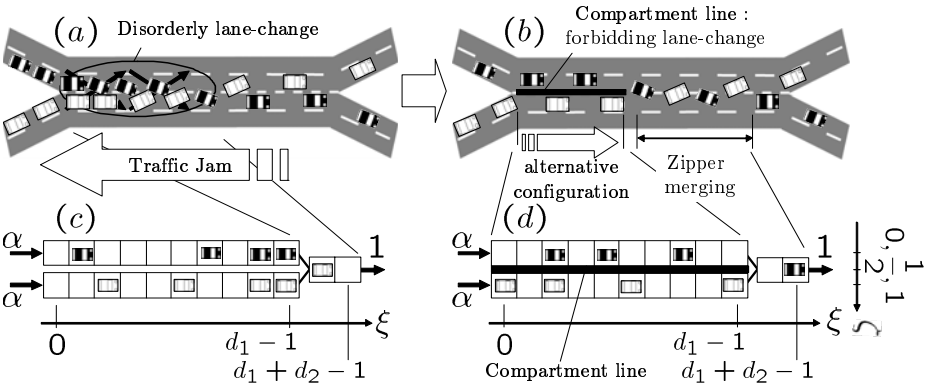


Fig. 1. (a) Disorderly lane-changes, which are caused by vehicles encountering suddenly with others, lead to traffic jams. (b) A compartment line drawn between two lanes makes zipper merging. Vehicles moving along this line see those on another lane and adjust their configurations to the alternative configurations. (c) A model of road (a) by using a two-lane lattice merging into a one-lane lattice. Vehicles on lattice (c) move irrespective of vehicles on another lane before merging. (d) A model of road (b). Vehicles on lattice (d) interact with others on another lane before merging.

2 Modeling

Here we define the dynamics of vehicles. We define $(\xi_i^t, \zeta_i^t) \in (\mathbb{Z}, \{0, 1/2, 1\})$ as the coordinates of the $i \in \mathbb{Z}$ th vehicle at time $t \in \{0, 1, 2, \dots\}$. $\zeta_i^t = 0$ and $\zeta_i^t = 1$ is the coordinate of vehicles on the two lanes, and $\zeta_i^t = 1/2$ denotes the

coordinate of vehicles changing lanes. The i th vehicle move at most one cell between time t and $t + 1$ with probability v_i^t as

$$(\xi_i^{t+1}, \zeta_i^{t+1}) = \begin{cases} (\xi_i^t + 1, \zeta_{i,aim}^t), & \text{with prob. } v_i^t, \\ (\xi_i^t, \zeta_i^t), & \text{with prob. } 1 - v_i^t. \end{cases} \tag{1}$$

$\zeta_{i,aim}^t \in \{0, 1/2, 1\}$ denotes the lane where the i th vehicle tries to stay at time $t + 1$, e.g., $\zeta_{i,aim}^t = \zeta_i^t$ denotes that it try to move straight, whereas $\zeta_i^t \in \{0, 1\}$ and $\zeta_{i,aim}^t = 1/2$ denote that it tries to change lanes.

The hopping probability v_i^t is given the following equation

$$v_i^t = \begin{cases} 0, & \Delta\xi_{same}(i, t) = 0, \\ r, & \Delta\xi_{same}(i, t) \geq 1, \text{ and } \Delta\xi_{anthr}(i, t) = 0, \\ q, & \Delta\xi_{same}(i, t) \geq 1, \text{ and } \Delta\xi_{anthr}(i, t) = 1, \\ 1, & \Delta\xi_{same}(i, t) \geq 1, \text{ and } \Delta\xi_{anthr}(i, t) \geq 2, \end{cases} \tag{2}$$

where $\Delta\xi_{same}(i, t)$ is the distance between the i th vehicle and the one in front of it occupying the same lane at time t , and $\Delta\xi_{anthr}(i, t)$ is the difference between the positions of the i th vehicle and the one in front on the nother lane at time t . $\Delta\xi_{same}(i, t)$ and $\Delta\xi_{anthr}(i, t)$ are given as

$$\Delta\xi_{same}(i, t) = \begin{cases} \xi_{s(i,t)}^t - \xi_i^t - 1, & \text{if the } s(i, t)\text{th vehicle exists,} \\ \infty, & \text{else,} \end{cases} \tag{3}$$

$$\Delta\xi_{anthr}(i, t) = \begin{cases} \xi_{a(i,t)}^t - \xi_i^t, & \text{if the } a(i, t)\text{th vehicle exists,} \\ \infty, & \text{else,} \end{cases} \tag{4}$$

where the $s(i, t)$ th vehicle is defined as the one in front of i th vehicle at time t which occupies the same lane with the i th vehicle, i.e., $|\zeta_{s(i,t)}^t - \zeta_i^t| \leq 1/2$ and the $a(i, t)$ th vehicle is defined as the one in front of i th vehicle at time t on another lane, i.e., $|\zeta_{a(i,t)}^t - \zeta_i^t| = 1$.

Vehicles obey the slow-to-start rule [18], which expresses the delay of acceleration caused by the inertia of vehicles. The hopping probability of the i th vehicle is not given as v_i^t but as sv_i^t under the condition that the i th vehicle are blocked by other one between time $t - 1$ and t and not blocked between time t and $t + 1$, i.e., $\Delta\xi_{same}(i, t - 1) = 0$ and $\Delta\xi_{same}(i, t) \geq 1$. $s \in [0, 1]$ is the parameter of slow-to-start rule. Small s denotes the large slow-to-start effect, i.e., the large inertia of vehicles. The model composed of (2) with SIS rule is named Multiple Lanes Slow-to-Start (MLSIS) model.

3 Simulations and Mean-Field Analysis

We investigate the effect of zipper merging induced by the compartment line. In this paper we treat for simplicity that all vehicles change lanes and vehicles

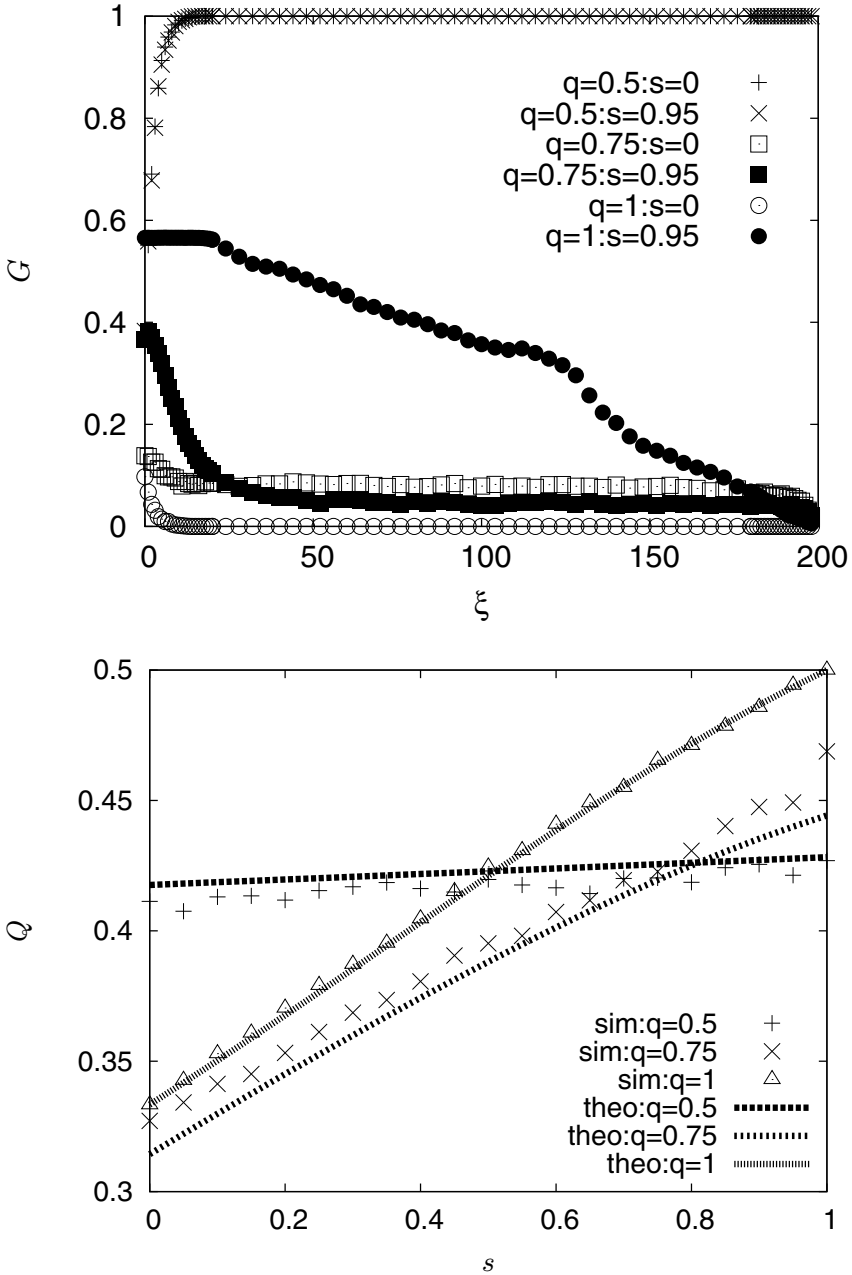


Fig. 2. G versus ξ (left) and Q versus s (right) for $q = r = 1$ (lattice (c)) and $q = r \in \{0.5, 0.75\}$ (lattice (d)). In the case of $q = r = 0.5$, G becomes 1 at $\xi = d_1 - 2$, and Q in this case intersects Q in other cases.

change lanes at the first cell where they are permitted. Thus, we have performed numerical simulations by using the lattices as shown in Fig. 1 (c-d). They are two-lane lattices merging into one-lane lattice. These lattices do not demand complicate rules of lane-changes in two-lane lattices, and hold the essence in merging, i.e., the bottleneck effects due to disorderly lane-changes. The coordinate is defined as $(\xi, \zeta) \in (\{0, 1, \dots, d_1 + d_2 - 1\}, \{0, 1/2, 1\})$, where $(d_1, 1/2)$ is the coordinate of the merging cell. The difference of lattice (c) and lattice (d) is the existence of the compartment line. Lattice (c) does not have it, on the other hand, lattice (d) has it at $0 \leq \xi \leq d_1$. Accordingly, vehicles on lattice (c) do not interact with others on the neighboring lane at $0 \leq \xi \leq d_1 - 1$, on the other hand, vehicles on lattice (d) do. The boundary conditions are given as follows. Vehicles try to enter in $(0, \zeta)$ ($\zeta \in \{1, 2\}$) with probability α when the cell is empty. Vehicles go out from $(1/2, d_1 + d_2 - 1)$ with probability 1. $\zeta_{i,aim}^t$ is given as $\zeta_{i,aim}^t = 1/2$ as long as $\xi_i^t = d_1 - 1$ and $\zeta_{i,aim}^t = \zeta_i^t$ for the other cases.

We measure flow rate Q versus s together with G versus ξ for $0 \leq \xi \leq d_1 - 2$, where G is defined as the degree of the alternative configurations of vehicles 14. The simulation conditions are given as follows. Vehicles are updated in parallel. The parameters are given as $\alpha = 0.33$, $s \in \{0, 0.05, \dots, 1\}$, $d_1 = 200$, and $d_2 = 10$. The period of the measurement is given as $10^4 \leq t \leq 2 \times 10^4 - 1$. Different q and r are given between lattice (c) and lattice (d). We use $q = r = 1$ for lattice (c), which denotes no interactions across the lanes at $0 \leq \xi \leq d_1 - 1$ and $q = r \in \{0.5, 0.75\}$ for lattice (d), which denotes that vehicles decrease their hopping probability in responding to others on the neighboring lane. Note that in this paper we do not treat the accelerations in responding to others because it increases the danger of collisions.

The results of the simulations are shown in Fig. 2. G becomes 1 at $\xi = d_1 - 2$ only in the case of $q = r = 0.5$, and we observe one crossing of Q between $q = r = 0.5$ and others. $G = 1$ denotes that zipper merging is achieved. Moreover, the crossing denotes that flux becomes larger (smaller) by achieving zipper merging with the hesitant interactions as long as s is small (large). Note that $G < 1$ in the case of $q = r = 0.75$ suggests that the length of the line is not enough.

In addition to the simulations, we also calculate the stationary flux by using mean field analysis. This theoretical flux agrees well with simulations as shown in Fig. 2.

4 Conclusive Discussions

It is shown that zipper merging is achieved by only the local hesitant interactions, and it is observed that the flow rate becomes larger in the zipper merging when inertia of vehicles are large. Moreover, theoretical flow rate coincides with simulation results. Our results are expected to be applied for the real traffic.

Acknowledgements

This work is supported by the Japan Society for the Promotion of Science and the Japan Science and Technology Agency. Two of the authors (Nishi and

Yanagisawa) were supported through the Global COE Program, “Global Center of Excellence for Mechanical Systems Innovation,” by the Ministry of Education, Culture, Sports, Science and Technology.

References

1. Helbing, D.: *Rev. Mod. Phys.* 73, 1067 (2001)
2. Chowdhury, D., Santen, L., Schadschneider, A.: *Phys. Rep.* 329, 199 (2000)
3. Chowdhury, D., Nishinari, K., Santen, L., Schadschneider, A.: *Stochastic Transport in Complex Systems: From Molecules to Vehicles*. Elsevier Science, Amsterdam
4. Bando, M., Hasebe, K., Nakayama, A., Shibata, A., Sugiyama, Y.: *Phys. Rev. E* 51, 1035 (1995)
5. Treiber, M., Hennecke, A., Helbing, D.: *Phys. Rev. E* 62, 1805 (2000)
6. Nagel, K., Schreckenberg, M.: *J. Physique I* 2, 2221 (1992)
7. Evans, M.R., Rajewsky, N., Speer, E.R.: *J. Stat. Phys.* 95, 45 (1999)
8. Nishinari, K., Fukui, M., Schadschneider, A.: *J. Phys. A* 37, 3101 (2004)
9. Kanai, M., Nishinari, K., Tokihiro, T.: *Phys. Rev. E* 72, 035102(R) (2005)
10. Kita, H.: *Transp. Res., Part A Policy Pract.* 33, 305 (1999)
11. Yamauchi, A., Tanimoto, J., Hagishima, A., Sagara, H.: *Phys. Rev. E* 79, 036104 (2009)
12. Hidas, P.: *Transp. Res., Part C Emerg. Technol.* 13, 37 (2005)
13. Davis, L.C.: *Physica A* 361, 606 (2006)
14. Nishi, R., Miki, H., Tomoeda, A., Nishinari, K.: *Phys. Rev. E* 79, 066119 (2009)
15. Kato, S., Tsugawa, S., Tokuda, K., Matsui, T., Fujii, H.: *IEEE Trans. on Intelligent Transportation Systems* 3, 155 (2002)
16. Ikemoto, Y., Hasegawa, Y., Fukuda, T., Matsuda, K.: Zipping, weaving control of vehicle group behavior in non-signalized intersection. In: *Proceedings. ICRA 2004*, vol. 5, p. 4387. IEEE, Los Alamitos (2004)
17. Takayasu, M., Takayasu, H.: *Fractals* 1, 860 (1993)
18. Benjamin, S.C., Johnson, N.F., Hui, P.M.: *J. Phys. A* 29, 3119 (1996)
19. Barlovic, R., Santen, L., Schadschneider, A., Schreckenberg, M.: *Eur. Phys. J. B* 5, 793 (1998)
20. Schadschneider, A., Schreckenberg, M.: *Ann. Physik* 6, 541 (1997)

Clustering and Transport Efficiency in Public Conveyance System

A. Tomoeda¹, R. Nishi^{2,3}, and K. Nishinari^{4,5}

¹ Meiji Institute for Advanced Study of Mathematical Sciences, Meiji Univ., Japan

² Department of Aeronautics and Astronautics Engineering, Univ. of Tokyo, Japan

³ Research Fellow of the Japan Society for the Promotion of Science, Japan

⁴ Research Center for Advanced Science and Technology, Univ. of Tokyo, Japan

⁵ PRESTO, Japan Science and Technology Corporation

atom@isc.meiji.ac.jp

http://dow.mydns.jp/

Abstract. Traffic efficiency of the public conveyance system on a given route is numerically and analytically investigated by introducing a new quantitative measure, so-called *transportation volume*, which is defined by the product of velocity and the number of on-board passengers. In terms of this measure, the optimal density of vehicles, at which the average velocity becomes maximum or the flow of particles becomes maximum, does not always make the transportation volume maximum. Moreover, under both with and without information-based control system, we have shown that this transportation volume shows some constant value almost everywhere in the density, even though the average velocity and the number of on-board passengers per unit bus decrease in the higher density of vehicles.

1 Introduction

Recently, various kinds of jamming phenomena, such as the dynamics of vehicular traffic, pedestrian flow and public transportations, are actively investigated from a point of view of statistical mechanics and non-equilibrium dynamics of interacting self driven particles [1,2,3]. Especially, the essential features of these jamming phenomena are well described as the extensions of the basic stochastic cellular automaton model such as the totally asymmetric simple exclusion process (ASEP) [4,5].

Until now, public conveyance traffic system such as buses, bicycles and trains have also been modeled by an extension of ASEP as the following similar approaches [6,7,8,9]. A simple bus route model [7] exhibits clustering of the buses along the route. The quantitative features of the coarsening of the clusters have strong similarities with coarsening phenomena in many other physical systems. Under normal circumstances, such clustering of buses is undesirable in any real bus route as the efficiency of the transportation system is adversely affected by clustering. In our previous study [8], a new public conveyance model (PCM) applicable to buses and trains is proposed by introducing the realistic effects of

the field, which are the number of stops and passengers behavior of getting on a vehicle at stops. In this model, we have discussed the efficiency of the bus route system which includes the hail-and-ride system and have found that the clustering of vehicles is quite similar to that observed in the ant-trail model [10,11]. Moreover, we have found that a big cluster of buses is divided into small clusters, by incorporating information of the number of vehicles between successive stops.

Now let us review our PCM for the bus operation system [8], that is, the particle and the field variable are buses and passengers at each bus-stops, respectively. The one-dimensional road is considered as a periodic and partitioned into L identical cells such that each cell can accommodate at most one particle at a time. A total of S ($0 \leq S \leq L$) equispaced cells is identified in the beginning as bus stops. At any given time step, a passenger arrives with probability f to the system. Here, we assume that a given passenger is equally likely to arrive at any one of the bus stops with probability $1/S$. Thus, the average number of passengers that arrive at each bus stop per unit time is given by f/S . The hopping probability H of each bus entering into next cell is defined by the form

$$H = \frac{Q}{\min(N_i, N_{\max}) + 1}, \quad (1)$$

where Q is the hopping probability without correlations between buses and passengers. N_i and N_{\max} are the number of waiting passengers at stop $i \in \{1, \dots, S\}$ and maximum boarding capacity of a bus, respectively. The form (1) is motivated by the common expectation that the time needed for the passengers embarking a bus is proportional to their number. Note that, this model reduces to ASEP in the case that H is constant such as $H = Q$. In principle, this hopping probability H in real bus operation system would also depend on the number of disembarking passengers. However, in order to keep the model theoretically simple and tractable, we ignore the situation that passengers get off only at those stops where waiting passengers to get into the bus and assume that the time taken by embarking passengers is always adequate for the disembarking passengers.

Furthermore, we introduce a traffic control system that exploits the information on the number of buses in each *segment* between successive bus stops, as a block section of railway system. Every bus stop has information I_i which is the number of buses in the i -th segment between i -th and next $i + 1$ -th bus stops. If I_i is larger than the average value $I_0 = m/S$, where m indicates the total number of buses, a bus remains stranded at a stop i as long as I_i exceeds I_0 . As shown in Fig. 1, the head distribution is dispersed by the effect of the information. The average headway distance with the information-based control system is much longer than that without control. Thus, the availability of the information I_j and implementation of the traffic control system based on this information, significantly reduce the undesirable clustering of buses.

In order to calculate the efficiency of the public conveyance system, we have also introduced two different quantitative measures of the efficiency, namely the average velocity of the vehicles $\langle V \rangle$ and the number of waiting passengers $\langle N \rangle$, and hence, an efficient system is considered as the higher $\langle V \rangle$ and lower $\langle N \rangle$. As shown in Fig. 2, one of the significant results in [8] shows that the

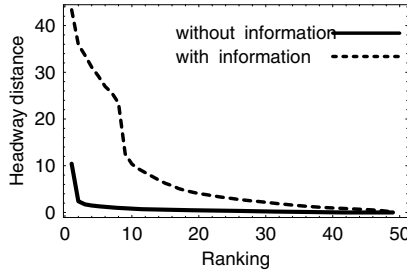


Fig. 1. The distribution plot of headway distance against the ranking in [8]

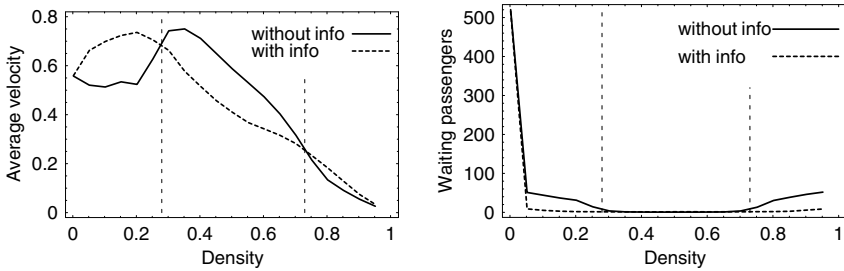


Fig. 2. Two efficiency plots $\langle V \rangle$, $\langle N \rangle$ in [8] for the parameters $S = 5$, $Q = 0.9$, $q = 0.5$, and $f = 0.9$

information-based traffic control system does not necessarily always improve the efficiency of the public conveyance system. In the middle density ($0.3 < \rho < 0.7$) the average velocity $\langle V \rangle$ is rather higher if the information-based control system is not executed. Unfortunately, however, the flow of the transportation system has hardly discussed in the previous study. Thus, the main aim of this paper is to discuss the flow of passengers, by introducing a new quantitative measure, so-called *transportation volume*.

2 Fundamental Diagrams in Public Conveyance System

In order to know the essences of traffic flow such as critical density, functional relation between the vehicle flow Q and the vehicle density ρ , which is called *fundamental diagram*, is mostly used. In this paper, we investigate two different kinds of the flow for the fundamental diagram, i.e., the flow of vehicles and the flow of carrying transportation volume. In this paper, we set $L = 500$, $S = 5$, $Q = 0.9$, $q = 0.5$, and $N_{\max} = 60$.

The former typical fundamental diagrams for the flow of vehicles in the public conveyance model are given in Fig. 3. The flow of vehicles without information-based control system gradually decreases as the arrival rate of passengers increases. In contrast, the flow with information-based control system drastically

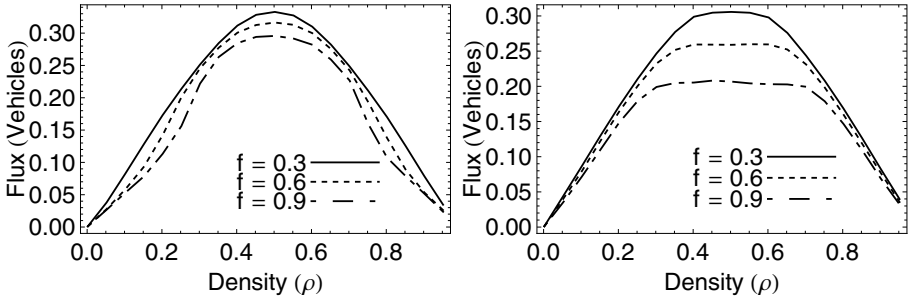


Fig. 3. Fundamental diagrams for the flow of vehicles. The left figure corresponds to the case without information-based control system and the right one corresponds to the case with information-based control system.

decreases in the middle density, where there are no waiting passengers in Fig. 2 and shows the trapezoidal shape. This trapezoidal shape is similar to the fundamental diagram in [12] and [13], where a blockage effect is artificially introduced into the rule-184 cellular automaton to take a flow bottleneck into account. Thus, under the absence of waiting passengers, implementation of the information-based control system corresponds to create the bottleneck effect and decreases the flow of vehicles.

Moreover, we introduce the transportation volume R as a new measure to estimate the efficiency of the system. The transportation volume R is defined by

$$R = \sum_{i=1}^m M_i V_i, \tag{2}$$

where $M_i (0 \leq M_i \leq N_{\max})$ and $V_i \in \{0, 1\}$ are the number of on-board passengers and the velocity of i -th bus, respectively.

Here, we have obtained another kind of the fundamental diagram which is shown in Fig. 4 by considering the flow as the transportation volume. Under the given arrival late f , the transportation volume of the system is maintained substantially constant except the low density limit in both cases, even though the density of vehicles increases. Note that in Fig. 4, the transportation volume at the lowest and highest density is not 0, since the lowest and highest density of numerical simulations is not $\rho = 0.0$ and 1.0 , but $\rho = 0.002$ and 0.952 , respectively. From the results, it is not always efficient for the system where the higher $\langle V \rangle$ and lower $\langle N \rangle$ in terms of transportation volume.

As shown in Fig. 2, the average velocity decreases in the region $\rho > 0.4$ (0.2) for the case without (with) information-based control system.

However, we have found that the number of buses with passengers at the high density $\rho = 0.502$ is more than that at the low density case $\rho = 0.202$ from Fig. 5, which shows the distribution of transportation volume against the ranking, where we arrange the order of magnitude according to the transportation volume of buses in descending order. Note that, the absence region for $\rho = 0.202$ in Fig. 5

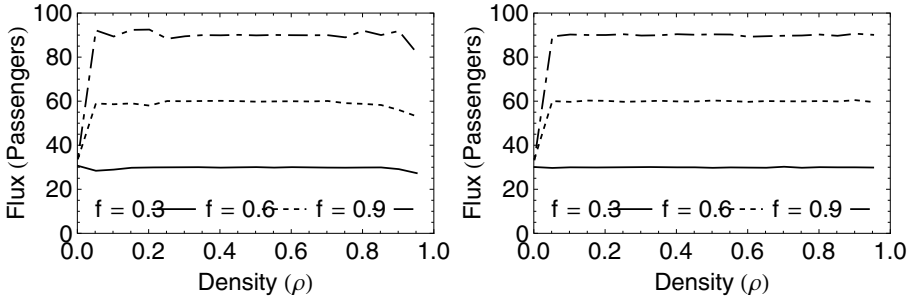


Fig. 4. Fundamental diagrams for the transportation volume. The left figure corresponds to the case without information-based control system and the right one corresponds to the case with information-based control system.

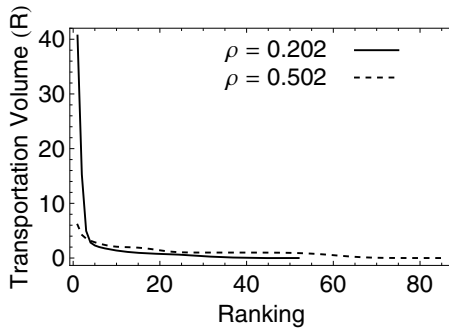


Fig. 5. Distribution of transportation volume against the ranking for the parameter $f = 0.9$ in the case without information-based control system

indicates the buses with no passengers. Therefore, the transportation volume is maintained substantially constant, since the number of buses with the small transportation volume increases even though the average velocity decreases. The number of empty buses in both cases is shown in Fig. 6. The diagonal line corresponds to the number of buses existing on a route based on its density. It is found that the number of empty buses increases as the density increases in both cases. Therefore, the difference between the diagonal line and the number of empty buses corresponds to the number of buses with passengers. The number of buses with passengers is quite similar in both cases except the high density. In the case without information-based control system, the transportation volume decreases, since the number of empty buses increases in Fig. 6, if the control system is switched off at the high density. Thus, the excess buses cause the inessential empty buses, since the transportation amount does not change even though the density increases.

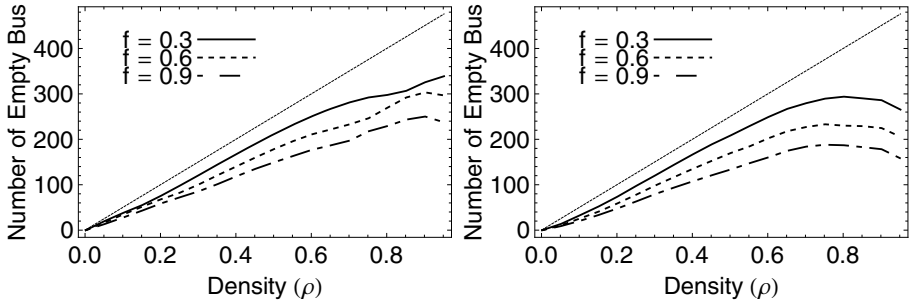


Fig. 6. The number of empty bus against the density for the parameter $f = 0.9$. The left figure corresponds to the case without information-based control system and the right one corresponds to the case with information-based control system.

Moreover, these results reveal that the transportation volume does not change between both conditions, where the information-based control system is switched on and off. That is, the transportation volume makes no difference between both cases with the clustering of vehicles and without the clustering.

3 Mean Field Analysis

Let us theoretically estimate the transportation volume R derived from $\langle V \rangle$ for the fully packed bus in the low density limit $\rho \rightarrow 0$. Suppose, T is the average time taken by a bus to complete one circuit of the route, denoted by L as noted before. The number of hops made by a bus with probability

$$q = \frac{Q}{N_{\max} + 1}, \tag{3}$$

during the average period T is S , and hence the time T for a bus is obtained by

$$T = \frac{L - S}{Q} + \frac{S}{q}. \tag{4}$$

Thus, the average velocity $\langle V \rangle$ is approximated by

$$\langle V \rangle = \frac{L}{T} = \frac{LQ}{L + SN_{\max}}. \tag{5}$$

Therefore, the transportation volume R for $m = 1$ is described as

$$R = N_{\max} \langle V \rangle = \frac{N_{\max} L Q}{L + S N_{\max}}. \tag{6}$$

For example, we have estimated $R = 33.8$ from the mean field approximation for the parameters $L = 500, Q = 0.9, N_{\max} = 60, S = 5$, and $m = 1$. The

corresponding value we have obtained directly from computer simulations is $R = 34.2$ (without information, $f = 0.9$) and $R = 33.5$ (with information, $f = 0.9$). This mean field estimation agrees almost perfectly with the corresponding simulation data, if the parameters satisfies the following condition

$$mSN_{\max} < fT, \quad (7)$$

where this condition signifies that the number of arrival passengers fT exceeds the maximum transporting capacity mSN_{\max} for m buses during time T . That is, the condition for arrival rate f is obtained by

$$f > \frac{mQSN_{\max}}{L + SN_{\max}}. \quad (8)$$

If arrival rate f does not satisfy this condition, it is not sufficient number of passengers in the system to pack a bus.

4 Conclusions and Acknowledgments

In this contribution, we have investigated two different kinds of the fundamental diagram for the public conveyance system. In the case of vehicles' flow, implementation of the information-based control system decreases the flow due to the bottleneck effect under the absence of waiting passengers. Whereas, in the case of the transportation volume, the optimal density which shows higher $\langle V \rangle$ and lower $\langle N \rangle$ does not always correspond to the most efficient operation for the transportation volume, since that is maintained substantially constant except the low density, even though the density of vehicles increases. This reason is considered that the number of buses with the small transportation volume increases even though the average velocity decreases. Thus, we have found that the excess buses cause the inessential buses which has no passengers since the transportation volume is same. In the near future, we would analytically exhibit that the transportation volume takes constant value.

Finally, the author (AT) is grateful to the Meiji University Global COE Program "Formation and Development of Mathematical Sciences Based on Modeling and Analysis" and the Ministry of Education, Culture, Sports, Science and Technology (MEXT) "Grant-in-Aid for Young Scientists (B)" for the support. The author (RN) acknowledges the support of the Japan Society for the Promotion of Science.

References

1. Chowdhury, D., Santen, L., Schadschneider, A.: Phys. Rep. 329, 199 (2000)
2. Helbing, D.: Rev. Mod. Phys. 73, 1067 (2001)
3. Chowdhury, D., Nishinari, K., Santen, L., Schadschneider, A.: Stochastic Transport in Complex Systems: From Molecules to Vehicles. Elsevier Science, Amsterdam (2010)

4. Derrida, B.: Phys. Rep. 301, 65 (1998)
5. Schutz, G.M.: Phase Transitions and Critical Phenomena, New York 19 (2001)
6. Jiang, R., Jia, B., Wu, Q.S.: J. Phys. A 37, 2063 (2004)
7. O'Loan, O.J., Evans, M.R., Cates, M.E.: Europhys. Lett. 42, 137 (1998); Phys. Rev. E 58, 1404 (1998)
8. Tomoeda, A., et al.: Physica A 384, 600 (2007)
9. Tomoeda, A., et al.: GESTS International Transaction on Computer Science and Engineering 54, 81 (2009)
10. Chowdhury, D., Guttal, V., Nishinari, K., Schadschneider, A.: J. Phys. A: Math. Gen. 35, L573 (2002)
11. Kunwar, A., John, A., Nishinari, K., Schadschneider, A., Chowdhury, D.: J. Phys. Soc. Jpn. 73, 2979 (2004)
12. Nishinari, K., Takahashi, D.: J. Phys. A: Math. Gen. 32, 93 (1999)
13. Yukawa, S., Kikuchi, M., Tadaki, S.: J. Phys. Soc. Japan 63, 3609 (1994)

Clusters in the Helbing's Improved Model

Rosa María Velasco¹ and Patricia Saavedra²

¹ Department of Physics

² Department of Mathematics,
Universidad Autónoma Metropolitana
Iztapalapa, 09340, México

Abstract. The formation of clusters in Helbing's improved model is studied by an iterative method. It is shown that after certain density we will always obtain a density profile which has the structure of a soliton. Its characteristics such as the amplitude and width are determined by the parameters in the model.

1 Introduction

The macroscopic traffic flow models represent a possible approach to study vehicle behavior in a highway. They are based on an analogy between compressible flow in a Navier-Stokes fluid and the traffic flow. In this work we have chosen the improved Helbing's model [1] which considers the continuity equation for the density $\rho(x, t)$, the equation describing the average speed $V(x, t)$ and, the speed variance equation $\Theta(x, t)$. This model introduced the length of vehicles as well as a safe distance between them and experimental information is used to calculate them. This work concerns the formation of traveling waves in a closed circuit.

2 The model

The model introduced by Helbing [1] is written in the conservative form

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{w})}{\partial x} = \mathbf{s}(\mathbf{w}), \quad (1)$$

where $\mathbf{w} = (\rho, \rho V, \rho \Theta)$,

$$\mathbf{F}(\mathbf{w}) = \begin{bmatrix} \rho V \\ \rho V^2 + P \\ \rho V \Theta + \lambda \frac{\partial \Theta}{\partial x} \end{bmatrix} \quad (2)$$

and

$$\mathbf{s}(\mathbf{w}) = \begin{bmatrix} 0 \\ \frac{(V_e(\rho) - V)}{\tau} \\ \frac{2(\Theta_e(\rho) - \Theta)}{\tau} - 2P \frac{\partial V}{\partial x} \end{bmatrix}. \quad (3)$$

The traffic pressure P is proposed with a viscosity coefficient η_0 and, a size correction for vehicles $s(V) = l + V\Delta T$ where $l = 7\text{ m}$ is the vehicle length and $\Delta T = 0.75\text{ s}$ is the reaction time. These assumptions drive to a traffic pressure given by

$$P(x, t) = \frac{\rho(x, t)\Theta(x, t)}{1 - \rho(x, t)s(V)} - \eta \frac{\partial V}{\partial x}. \tag{4}$$

The coefficients $\eta = \eta_0/(1 - \rho s)$ and $\lambda = \lambda_0/(1 - \rho s)$ contain the size correction and, τ is the relaxation time. On the other hand the speed $V_e(\rho)$ corresponds to the fundamental diagram

$$\frac{V_e}{V_{max}} = -3.72 \times 10^{-6} + \left[1 + \exp\left(\frac{\frac{\rho}{\rho_{max}} - 0.25}{0.06}\right) \right]^{-1}, \tag{5}$$

and $\Theta_e(\rho) = A(\rho)V_e(\rho)^2$, where $A(\rho)$ is the variance prefactor given in terms of experimental data correlations [2].

3 Stability

The homogeneous steady state $\mathbf{w}_e = (\rho_e, V_e(\rho_e), \Theta_e(\rho_e))$ is a solution of the equations of motion and a small perturbation around it will tell us the conditions for stability. The perturbation is given through $\mathbf{w} = \mathbf{w}_e + \tilde{\mathbf{w}}\exp(ikx + \gamma t)$ and the dispersion relation allows the calculation of roots. Then, when $\Re \gamma < 0$ the solution will be stable. There are three values of such quantity and the final condition is obtained taking the lowest order in the wave vector k , it is given as

$$(\rho_e V_e')^2 \leq \alpha \Theta_e(1 + \alpha \rho_e s_e) + \alpha^2 \rho_e^2 \Theta_e \Delta T + \alpha \rho_e \Theta_e', \tag{6}$$

a result which implies that the model is linearly stable for densities lower than 11.73 veh/km . The unstable region produces a nonhomogeneous profile with the soliton characteristics.

4 The Soliton Structure

The presence of a soliton solution in this model is exhibited by means of writing the equations of motion in a moving reference frame $z = x - c_s t$. An iterative method is applied taking into account the existence of two different time scales, one is given through τ and the other one is carried with the stability condition [6] [3]. The equation describing the profile is the well known Korteweg-de Vries equation,

$$\frac{\eta_0 \tau \alpha (V_e - c_s)}{\rho_e (c - c_s)} \rho_{zzz} + \frac{\beta}{(c - c_s)} \hat{\rho} \hat{\rho}_z + \hat{\rho}_z = 0, \tag{7}$$

its solution is given as

$$\hat{\rho} = \frac{3(c_s - c)}{2V_e' + \rho_e V_e''} \operatorname{sech}^2 \left[\frac{1}{2} \sqrt{\frac{\rho_e}{\eta_0 \tau \alpha} \frac{c - c_s}{V_e - c_s}} z - z_0 \right], \tag{8}$$

where $c = V_e(\rho_e) + \rho_e V_e'(\rho_e)$ is the propagation speed of long wavelength perturbations, $\alpha = 1 - \rho_e s_e$ and, all primes indicate derivative with respect to the density. The soliton amplitude and its width are given as

$$B = \frac{3(c_s - c)}{2V_e' + \rho_e V_e''}, \quad D = \frac{\pi^2}{12} \sqrt{\frac{4\eta_0 \alpha \tau}{\rho_e} \frac{(V_e - c_s)}{(c - c_s)}}. \tag{9}$$

Both are determined by the parameters in the model.

5 Simulation

We solve the nonlinear equations (2) and (3), with periodic boundary conditions and take highway length as $L = 6 \text{ km}$. Concerning the initial conditions we have taken an homogeneous traffic state $\rho_e = 28 \text{ veh km}^{-1}$ in the unstable region plus a small perturbation on the density, as the one given by (10) with $C_1 = C_2 = 4 \text{ veh km}^{-1}$, $\omega_1 = \omega_2 = 0.5$, $x_0 = -3.0 \text{ km}$ and $x_1 = 3.0 \text{ km}$,

$$\rho(x, 0) = \rho_e + C_1 \cosh^{-2}\left(\frac{x - x_0}{\omega_+}\right) - C_2 \frac{\omega_+}{\omega_-} \cosh^{-2}\left(\frac{x - x_1}{\omega_-}\right), \tag{10}$$

$$\rho(x, 0)V(x, 0) = \rho_e V_e(\rho_e), \quad \Theta_e(\rho(x, 0)) = \Theta_e(\rho_e). \tag{11}$$

The figures (1,2) give us a traveling profile which moves with speed c_s in the opposite direction of vehicles. Figure (3) represents the profile obtained for a bigger density ρ_e given the same initial conditions.

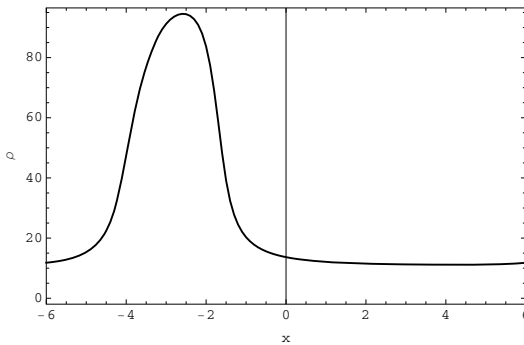


Fig. 1. Density profile at $t = 100 \text{ min}$

6 Concluding Remarks

The Helbing's improved model shows a permanent profile with soliton structure. It is slightly asymmetric, its amplitude and width depend on the parameters in the model. Figures (1,3) show clearly that the density ρ_e plays an important role. This kind of structure represents a traveling cluster moving with constant speed. Some other macroscopic models have similar properties [3], [4], [5], [6].

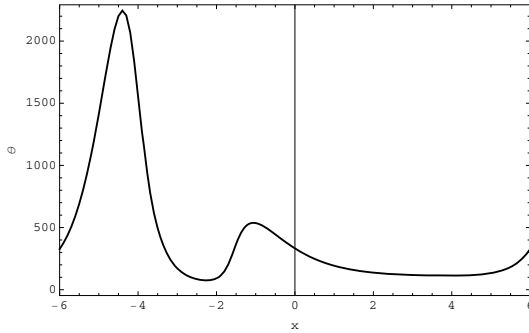


Fig. 2. Speed variance profile at $t = 100 \text{ min}$

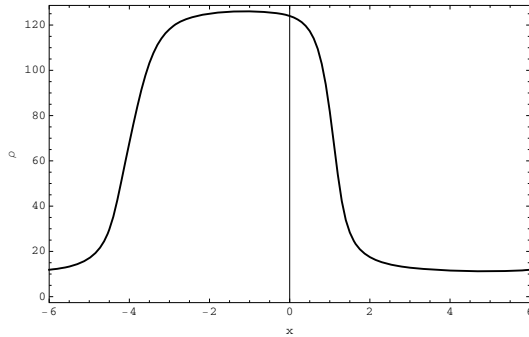


Fig. 3. Density profile when $\rho_e = 60$ at $t = 100 \text{ min}$. Observe the width.

References

1. Helbing, D.: Phys. Rev. E 53, 2366 (1996)
2. Treiber, M., Helbing, D.: J. Phys. A: Math. Gen. 32, L17 (1999)
3. Berg, P., Woods, A.: Phys. Rev. E 64, 035602(R) (2001); Berg, P., Mason, A., Woods, A.: Phys. Rev. E 61, 1056 (2000)
4. Kürtze, D.A., Hong, D.C.: Phys. Rev. E 52, 218 (1995)
5. Kerner, B.S., Konhäuser, P.: Phys. Rev. E 48, R2335 (1993); *ibid.* Phys. Rev. E 50, 54 (1994)
6. Saavedra, P., Velasco, R.M.: 12th IFAC Symposium on Transportation Systems CTS 2009, p. 428 (2009)

Phase Transitions in Cellular Automata for Cargo Transport and Kinetically Constrained Traffic

Marko Woelki

Theoretische Physik, Universität des Saarlandes,
66123 Saarbrücken, Germany
woelki@lusi.uni-sb.de

Abstract. A probabilistic cellular automaton for cargo transport is presented that generalizes the totally asymmetric exclusion process with a defect from continuous time to parallel dynamics. It appears as an underlying principle in cellular automata for traffic flow with non-local jumps for the kinetic constraint to drive as fast as possible. The exactly solvable model shows a discontinuous phase transition between two regions with different cargo velocities.

Keywords: asymmetric exclusion process, matrix-product state.

1 Introduction

Non-equilibrium phase transitions can rarely be calculated exactly, i.e. without need of approximations or fits of numerical data. One paradigmatic system where this is possible is the totally asymmetric simple exclusion process (TASEP) (see [1] and references therein). The model is defined on a 1d discrete lattice with sites being either empty (holes) or occupied by a single particle. A randomly chosen particle moves to the right at rate 1 provided that the site is empty. For open boundaries with particle input at rate α and output at rate β this leads to three different phases: a low-density, a high-density and a maximum-current phase. For finite systems the process can be solved exactly by the matrix-product Ansatz (see [2] for a recent and exhaustive review) and the complete thermodynamic behavior that is relevant for understanding the phase diagram can be extracted from the asymptotics of this solution. On the ring the process has a uniform groundstate [1]. However the presence of a defect particle leads to a rich phase behavior [3]. In the defect TASEP usual particles move $10 \rightarrow 01$ at rate 1, the single defect particle moves itself forward $20 \rightarrow 02$ at rate α and can be overtaken by usual particles $12 \rightarrow 21$ with β . The solution is formally related to the open-boundary case. There is one shock phase and three phases where it behaves once like a particle, once like a hole and once like a second-class particle. The second-class particle case corresponds to $\alpha = \beta = 1$. To the left the second-class particle looks like a hole (as seen from a particle) and to the right it looks like a particle (as seen from a hole). This case was studied in [4] since it can be used

to study the shock in the TASEP on the infinite line with step-initial condition. The connection is that, due to its very special dynamics, the defect 2 samples the random shock position on preferring configurations like 000021111.

The defect TASEP can alternatively be understood as a cargo transport process: The defect is a usual particle carrying a cargo which can be handed over to a particle behind. See [5] for a different definition of cargo in the same context. These mechanisms play a fundamental role in the biology of intracellular transport. Processive motors like kinesin transport cargo over long distances [6]. The cargo can alternatively be interpreted as virus particles that use carrier particles in order to attain the interior of a cell, see [5] for further references.

The Nagel-Schreckenberg model [7] is often referred to as the minimal model for one-lane traffic-flow on a freeway. There it is essential to allow for faster and slower cars to get a realistic flow-density relation: cars can move up to v_{max} sites per time step. Further all cars are updated simultaneously according to a parallel update and move independently with probability p . For $v_{max} = 1$ it is equivalent to the TASEP with parallel dynamics. The steady state on the ring shows nearest-neighbor correlations and has a simple pair-factorized form [8]. For open boundaries the matrix-product technique could be generalized to obtain the exact steady state [9,10]. It can be interpreted as a pair-factorized state as on the ring modulated by a matrix-product state [11].

In this article we introduce a generalization of the cargo-transport process to discrete time with parallel updating and give its exact solution. Here we restrict ourselves to light cargo, i.e. the case where the speed of a particle is not lowered by the presence of cargo. This appears naturally in the steady state of a traffic cellular automaton [11]. We will see that non-local jump processes where particles drive as fast as possible can lead for non-deterministic hopping to a discontinuous phase transition on the ring.

2 The Cellular-Automaton Model and Its Solution

Consider a periodic one-dimensional lattice with sites being either occupied by a particle (in state $\tau = 1$) or empty ($\tau = 0$). One of the particles carries a cargo to which we refer to as a defect ($\tau = 2$). The particles are updated simultaneously and every particle (with or without cargo) moves forward with probability p . If the site behind it is occupied, the cargo carrying particle can independently give its cargo back at probability β . This simple dynamics is encoded in detail in the transitions

$$10 \rightarrow 01, \text{ at rate } p, \tag{1}$$

$$020 \rightarrow x02, \text{ at rate } p, \tag{2}$$

$$120 \rightarrow 210, \text{ at rate } \beta(1-p), \tag{3}$$

$$\rightarrow 102, \text{ at rate } (1-\beta)p, \tag{4}$$

$$\rightarrow 201, \text{ at rate } \beta p, \tag{5}$$

$$121 \rightarrow 21x, \text{ at rate } \beta, \tag{6}$$

with x being either 0 or 1 indicating that the site can be either empty or occupied due to the parallel update. For example the evolution of the pattern 020 can be affected by a particle to the left moving itself forward. This is quite a general scenario that might apply to biological intracellular cargo transport. The parallel update reflects highly active transport where many particles move at the same time.

For $\beta = 0$ the cargo is attached to one special particle for all times and its dynamics is the same as for the other particles. This corresponds to the usual TASEP (with parallel update) and the single occupation $\tau = 2$ can be replaced by $\tau = 1$. The steady state for a lattice with $L + 1$ sites (with one of them occupied by the defect) is

$$P(\tau_1, \tau_2, \dots, \tau_{L+1}) = \prod_{i=1}^{L+1} P(\{\tau_{i-1}, \tau_i\}), \tag{7}$$

thus it factorizes [8] into symmetric two-site factors $P(\tau_{i-1}\tau_i) \equiv P(\{\tau_{i-1}, \tau_i\})$ with

$$P(00) = 1 - \rho - J/p, \tag{8}$$

$$P(10) = J/p, \tag{9}$$

$$P(11) = \rho - J/p. \tag{10}$$

Here J is the particle current

$$J(\rho) = \frac{1 - \sqrt{1 - 4p\rho(1 - \rho)}}{2}. \tag{11}$$

We found [11] that the steady state for $\beta > 0$ can be calculated exactly too. Here [7] is generalized to

$$P(2, \tau_1, \dots, \tau_L) \propto \tilde{f}(\tau_1)f(\tau_1\tau_2)\dots f(\tau_{L-1}\tau_L)\tilde{f}(\tau_L) \times \langle W | \left[\prod_{i \geq 1} \tau_i D + (1 - \tau_i) E \right] | V \rangle \tag{12}$$

This is a pair-factorized state (mainly the steady state for $\beta = 0$) modulated by a matrix-product state. Up to the normalization this is very related to the TASEP with open boundaries [9,11]. The vectors $\langle W |$ and $| V \rangle$ represent the defect and the matrices D and E represent particles and holes respectively. The operators obey the algebra

$$\langle W | EE = (1 - p)\langle W | E, \tag{13}$$

$$\langle W | ED = (1 - p)(\langle W | D + p), \tag{14}$$

$$DE = (1 - p)[D + E + p], \tag{15}$$

$$D|V\rangle = \frac{p(1 - \beta)}{\beta}|V\rangle, \tag{16}$$

for details see [11]. Note that the relations (15) and (16) are quadratic and the other relations are cubic. Accordingly the dynamical rules in (16) are quadratic (for 10 and 12) and cubic (for 20) respectively. The thermodynamic particle current (11) obviously is unaffected by the cargo.

This steady state appears also in cellular-automaton models for traffic flow with non-local jumps under kinetic constraint. Consider the following process on a periodic one-dimensional lattice with sites being either occupied by one car (in state $\tau = 1$) or empty ($\tau = 0$). The update rules applied simultaneously to all cars (\equiv particles) are

$$\begin{aligned} 100 &\rightarrow 001, & \text{with probability } p, \\ 101 &\rightarrow 01x, & \text{with probability } \beta, \end{aligned}$$

where x denotes either a particle or hole. The maximum velocity v_{\max} thus is two sites per time step instead of one in the usual TASEP and the kinetic constraint is that cars can not drive at reduced speed 1 if they could move at maximum speed. This leads under the parallel dynamics to a non-local repulsion between cars, so that finally in the thermodynamic limit only even gaps (0, 2, 4, ... holes) have non-vanishing probability. Figure 1 a) shows schematically the allowed moves in a stationary configuration. For even number of holes the process is equivalent to the TASEP and for odd number of holes it is equivalent to the cargo-transport process. In this case a single hole in an environment of particles and hole pairs is formed that plays the role of cargo attached to varying particles. In figure 1 b) one sees that the particle movement of a single site is equivalent to backward movement of the 01 position. The 01 pair plays the role of the defect, having the characteristic ‘Janus face’, looking to the left like a hole and to the right like a particle. Usual holes are replaced by $P(00) \equiv P(0000)$, $P(01) \equiv P(001) \equiv P(0001)$. In the following we restrict ourselves in the terminology to the cargo-transport process.

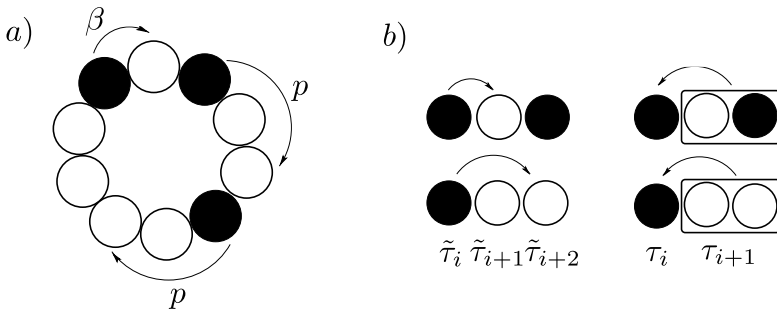


Fig. 1. a) Allowed moves in a stationary configuration: only one odd gap between particles b) Equivalence of moves and according reduction of lattice units

3 The Phase Diagram

There is a discontinuous phase transition at ρ_c the density

$$\rho_c = \frac{\beta(1 - \beta)}{p - \beta^2}. \tag{17}$$

For $\rho < \rho_c$ and $\rho > \rho_c$ one finds different velocities of the defect which can be calculated through

$$v = p(1 - \rho_+)(1 - \beta\rho_-) - \beta\rho_-. \tag{18}$$

Here ρ_- and ρ_+ are the densities directly behind and in front of the defect. The dynamics of the defect is obtained from (16). It moves either forward with probability p if it has a hole in front while at the same time there is no particle directly behind that simultaneously catches the cargo: first term in (18). Or it moves backwards with probability β if it has a particle behind: second term in (18). The neighboring densities are in terms of the current $J(\rho)$ defined in (11):

$$\rho_- = \begin{cases} \frac{1}{\beta(1 - \rho)^2} \frac{J^2(\rho - J)^2}{p(\rho - J)^2 + (1 - p)J^2}, & \text{for } \rho < \rho_c, \\ \frac{p\rho - J}{p\rho - \beta J} & \text{for } \rho > \rho_c, \end{cases} \tag{19}$$

$$1 - \rho_+ = \begin{cases} \left(\frac{J}{p\rho}\right)^2, & \text{for } \rho < \rho_c, \\ \frac{p - \beta}{p^2(1 - \beta)} \frac{J}{\rho} & \text{for } \rho > \rho_c. \end{cases} \tag{20}$$

Note that there are many ways to express the results due to the relation $J(1 - J) = p\rho(1 - \rho)$. In other words the square root in J appears in every power of J with a certain prefactor. Equations (19,20) yield

$$\frac{v(\rho)}{p} = \begin{cases} \frac{1 - 2\rho}{1 - 2J}, & \text{for } \rho < \rho_c, \\ \frac{J - \beta\rho}{p\rho - \beta J} & \text{for } \rho > \rho_c. \end{cases} \tag{21}$$

The defect velocity vanishes for

$$\rho_0 = \begin{cases} \frac{p - \beta}{p - \beta^2}, & \text{for } \beta < 1 - \sqrt{1 - p}, \\ 1/2, & \text{for } \beta > 1 - \sqrt{1 - p}, \end{cases} \tag{22}$$

and is positive for $\rho > \rho_0$ and negative for $\rho < \rho_0$. This leads to the phase diagram, depicted in figure 2. The formulae (17) and (22) can alternatively be interpreted in terms of a critical value $\beta_c \equiv \beta(\rho_c)$ and $\beta_0 \equiv \beta(\rho_0)$. This gives

$$\beta_c = J(\rho_c)/(1 - \rho_c), \quad \beta_0 = J(\rho_c)/\rho_c. \tag{23}$$

The value of β_c and β_0 respectively then is essentially the absolute velocity of holes and particles at the transition. Figure 3 shows the character of the defect

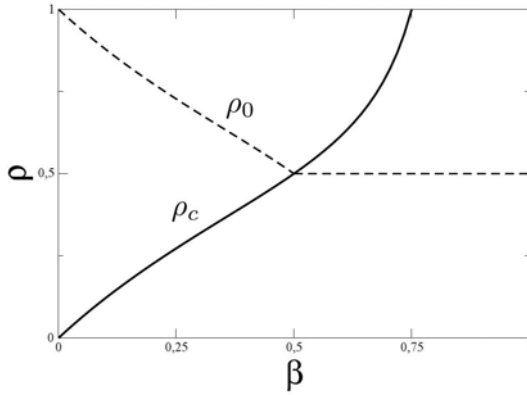


Fig. 2. The phase diagram shows the ρ - β plane for $p = 3/4$. The thick line separates the two phases and on the dashed line the velocity of the defect changes its sign.

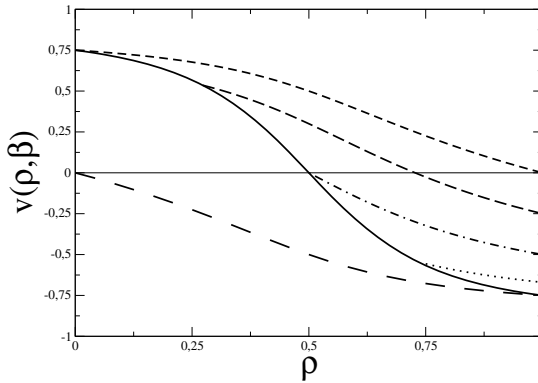


Fig. 3. Velocity of the defect for $p = 3/4$ for varying β . *Continuous line:* result for $\rho < \rho_c(\beta)$. The *broken lines* correspond to $\beta = 0, \beta = 0.25, \beta = 0.5, \beta = 0.67$ from top (*dashed*) to bottom (*dotted*) and are valid for $\rho > \rho_c(\beta)$. The *segmented line* is the velocity of holes.

and the discontinuous phase transition. For $\beta = 0$ its velocity is given by the upper curve and equals the velocity of particles. In the zero-density limit the velocity is independently of β equal to p . For increasing β the second phase appears: The velocity of the defect jumps at $\rho = \rho_c(\beta)$ from the continuous curve to the corresponding dashed curve. ρ_c increases with β until $\beta = p$ where $\rho_c = 1$, so that the system is completely in the second phase for all densities. Note that for $\beta = 1/2$ one has $\rho_c = 1/2$. Finally in the limit of the fully occupied lattice $\rho = 1$ and $\rho_c \leq 1$ the velocity equals β for $\beta \leq p$. However for $\rho_c > 1$ it can never increase the value of p which there is the corresponding velocity of the holes given by the segmented line.

Table 1. Continuous-time limit $p \equiv dt$, $\beta \equiv \tilde{\beta}dt$: Values to the order $\mathcal{O}(dt)$ of velocity, empty-space density in front and particle density behind the defect in the two phases. Here $\rho_c \sim \beta [1 - \beta(1 - \beta)dt]$.

ρ	v	$1 - \rho_+$	ρ_-
$\rho < \rho_c$	$(1 - 2\rho)dt$	$(1 - \rho)^2 [1 + 2\rho(1 - \rho)dt]$	$\rho^2/\beta [1 - (1 - \rho)(1 - 3\rho)dt]$
$\rho > \rho_c$	$(1 - \tilde{\beta} - \rho)dt$	$(1 - \tilde{\beta})(1 - \rho) [1 + (\tilde{\beta} + \rho(1 - \rho))dt]$	$\rho [1 - (1 - \rho)(1 - \tilde{\beta} - \rho)dt]$

For the phase $\rho < \rho_c$, which is purely present for $\beta \geq p$, it is important to stress that the quantities ρ_+ , $\beta\rho_-$ and v are independent of β . The density profile is symmetric around the defect and has an algebraic decay. This corresponds to the second-class particle phase in the defect TASEP mentioned in the introduction. As in continuous time the defect velocity (21) is given by $v = dJ/d\rho$ which has the form of a group velocity and becomes $v = 1 - 2\rho$ for small p , compare table 1. Thus the defect travels with the velocity of the density disturbance.

4 Limits

Table (1) shows the limit of small hopping probabilities: $p \equiv dt$, $\beta = \tilde{\beta}dt$ with $dt \rightarrow 0$. Note in comparison that the velocity of normal particles is always J/ρ which gives $p(1 - \rho) + p^2(1 - \rho)^2 + \dots$. This limit has been studied in the traffic picture in [12,13] and corresponds to the defect TASEP. As mentioned before, in the limit $\beta = 0$ the defect moves only forward and loses its role as a defect. Therefore the steady state is the same as for the TASEP. Thus one has the expressions given in table (2). In comparison the results from (19,20,21) for $\rho > \rho_c$ are rewritten and expanded around the TASEP value:

$$\rho_-(\beta) = \frac{P(11)}{\rho - \beta P(10)} = \frac{P(11)}{\rho} \left[1 + \frac{P(10)}{\rho} \beta + \dots \right] \tag{24}$$

$$1 - \rho_+(\beta) = \frac{p - \beta}{p(1 - \beta)} \frac{P(10)}{\rho} = \frac{P(10)}{\rho} \left[1 - \frac{1 - p}{p} \beta - \dots \right] \tag{25}$$

$$v(\beta) = \frac{J - \beta\rho}{\rho - \beta P(10)} = \frac{J}{\rho} - \left(1 - \frac{J^2}{p\rho^2} \right) \beta - \dots \tag{26}$$

One sees that ρ_- is mainly the same as for $\beta = 0$ but the density to which the numerator is addressed is reduced by backward moving so that ρ_- is increased. The same holds for the velocity v . $1 - \rho_+$ is even the same as for $\beta = 0$ up to a scale which is, using (17) and (22), given by ρ_c/ρ_0 .

Table 2. Limit of the discrete-time TASEP $\beta = 0$

v	$1 - \rho_+$	ρ_-
$\frac{P(10)}{\rho} = \frac{J}{\rho}$	$\frac{P(10)}{\rho} = \frac{J}{p\rho}$	$\frac{P(11)}{\rho} = 1 - \frac{J}{p\rho}$

For $p = 1$ (24)-(26) yield the results displayed in table 3. For $\rho < 1/2$ all particles are separated and move deterministically as in the TASEP. For $\rho > 1/2$ the effect of β on the velocity remains for all possible values and the phase transition disappears.

Table 3. The partially deterministic case $p = 1$: Velocity, empty-space density in front and particle density behind the defect

ρ	v	$1 - \rho_+$	ρ_-
$\rho < 1/2$	1	1	0
$\rho > 1/2$	$\frac{1 - \rho - \beta\rho}{\rho - \beta(1 - \rho)}$	$\frac{1 - \rho}{\rho}$	$\frac{2\rho - 1}{\rho - \beta(1 - \rho)}$

5 Conclusions

A cellular automaton for cargo transport was introduced that generalizes the (continuous-time) defect TASEP. The parallel update is often more realistic in describing active many-particle transport and makes the link between deterministic and random-sequential dynamics. The point of interest was a single defect, i.e. a particle carrying light cargo in an environment of particles and holes on a periodic 1d lattice. Particles move forward with probability p and if a particle is directly behind the particle that carries the cargo it may catch the cargo with probability β . We found a discontinuous phase transition between two phases with different cargo velocities. Successively increasing β lowers its velocity only until $\beta = p$. Then a saturation effect appears where the velocity becomes independent of β . The same holds for the stationary state of a cellular automaton for traffic where the ‘cargo’ corresponds to small headway that is formed dynamically. It is attached to one car from behind. If the subsequent car comes close enough it will catch it up. The fact that the cargo process appears in a seemingly unrelated non-local jump process underlines its universal role. The exact matrix-product state and its cubic algebra holds also for the two-species case where multiple cargo is present. It is also trivially generalized to case where cargo lowers the speed of the particle to $\alpha < p$. For $p = \beta$ and in the presence of several second-class particles it serves also as a model system for the study of shocks on the infinite line.

Acknowledgements

It is a pleasure to thank Kirone Mallick for his kind hospitality at IPhT.

References

1. Derrida, B.: The asymmetric simple exclusion process: An exactly soluble nonequilibrium system. Phys. Rep. 301, 65 (1998)
2. Blythe, R.A., Evans, M.R.: Nonequilibrium steady states of matrix product form: A solver’s guide. J Phys. A 40, R333 (2007)

3. Mallick, K.: Shocks in the asymmetry exclusion model with an impurity. *J. Phys. A* 29, 5375–5386 (1996)
4. Derrida, B., Janowsky, S.A., Lebowitz, J.L., Speer, E.R.: Exact solution of the asymmetric exclusion process: Shock profiles. *J. Stat. Phys.* 73(5/6), 813–843 (1993)
5. Goldman, C., Sena, E.: The dynamics of cargo driven by molecular motors in the context of asymmetric simple exclusion processes. *Physica A* 388, 3455–3464 (2009)
6. Korn, C., Klumpp, S., Lipowsky, R., Schwarz, U.S.: Stochastic simulations of cargo transport by processive molecular motors. *J. Chem. Phys.* 131, 245107 (2009)
7. Nagel, K., Schreckenberg, M.: A cellular automaton model for freeway traffic. *J. Phys. I France* 2, 2221–2229 (1992)
8. Schreckenberg, M., Schadschneider, A., Nagel, K., Ito, N.: Discrete stochastic models for traffic flow. *Phys. Rev. E* 51(4), 2939–2949 (1995)
9. Evans, M.R., Rajewsky, N., Speer, E.R.: Exact solution of a cellular automaton for traffic. *J. Stat. Phys.* 95, 45–96 (1999)
10. de Gier, J., Nienhuis, B.: Exact stationary state for an asymmetric exclusion process with fully parallel dynamics. *Phys. Rev. E* 59, 4899–4911 (1999)
11. Woelki, M., Schreckenberg, M.: Exact matrix-product states for parallel dynamics: Open boundaries and excess mass on the ring. *J. Stat. Mech.*, P05014 (2009)
12. Woelki, M., Schreckenberg, M.: Headway oscillations and phase transitions for diffusing particles with increased velocity. *J. Phys. A* 42, 325001 (2009)
13. Woelki, M., Schreckenberg, M.: Phase transitions and even/odd effects in asymmetric exclusion models. In: Appert-Roland, C., et al. (eds.) *Traffic and Granular Flow 2007*, pp. 435–440. Springer, Heidelberg (2008)

A New Computational Methodology Using Infinite and Infinitesimal Numbers

Yaroslav D. Sergeyev

Università della Calabria, 87030 Rende (CS), Italy and
N.I. Lobatchevsky State University, Nizhni Novgorod, Russia

yaro@si.deis.unical.it

<http://wwwinfo.deis.unical.it/~yaro>

Abstract. Traditional computers work numerically only with finite numbers. Situations where the usage of infinite or infinitesimal quantities is required are studied mainly theoretically. In this lecture, a new computational methodology (that is not related to non-standard analysis approaches) is described. It is based on the principle ‘The part is less than the whole’ applied to all quantities (finite, infinite, and infinitesimal) and to all sets and processes (finite and infinite). The new methodology has allowed the author to introduce the Infinity Computer working numerically with infinite and infinitesimal numbers. The new computational paradigm both gives possibilities to execute computations of a new type and simplifies fields of Mathematics and Computer Science where infinity and/or infinitesimals are required. Examples of the usage of the introduced computational tools are given during the lecture.

Keywords: Numeral systems, infinite and infinitesimal numbers, Infinity Computer, numerical computations, Turing machine.

There exist different ways to generalize traditional arithmetic for finite numbers to the case of infinite and infinitesimal quantities (see, e.g., [1,2,4,5] and references given therein). However, arithmetics that have been developed so far to deal with infinite quantities are quite different with respect to the finite arithmetic we are used to work with. In fact, arithmetics working with infinity can have undetermined operations (for example, $\infty - \infty$, $\frac{\infty}{\infty}$, etc.) or they use representation of infinite numbers based on infinite sequences of finite numbers. These difficulties did not allow people to create computers working with infinite and infinitesimal quantities numerically.

In this lecture, we describe a new methodology (see survey [8] and applications in [6,7,9,11,14,15]) for treating infinite and infinitesimal quantities expressed in a new numeral¹ system. It has a strong numerical character and is based on the principle ‘The part is less than the whole’ applied to all numbers (finite,

¹ We remind that *numeral* is a symbol or group of symbols that represents a *number*. A *number* is a concept that a *numeral* expresses. The same number can be represented by different numerals. For example, the symbols ‘8’, ‘eight’, and ‘VIII’ are different numerals, but they all represent the same number.

infinite, and infinitesimal) and to all sets and processes (finite and infinite). The new methodology has allowed the author to introduce the Infinity Computer (see European patent [12]) working numerically with infinite and infinitesimal numbers. The new computational paradigm both gives possibilities to execute computations of a new type and simplifies fields of Mathematics and Computer Science where infinity and/or infinitesimals are required.

In order to understand how it is possible to look at the problem of infinity in a new way, let us consider a study published in *Science* by Peter Gordon (see [3]) where he describes a primitive tribe living in Amazonia - Pirahã - that uses a very simple numeral system for counting: one, two, many. For Pirahã, all quantities bigger than two are just 'many' and such operations as $2+2$ and $2+1$ give the same result, i.e., 'many'. Using their weak numeral system Pirahã are not able to see, for instance, numbers 3, 4, 5, and 6, to execute arithmetical operations with them, and, in general, to say anything about these numbers because in their language there are neither words nor concepts for that. Moreover, the weakness of their numeral system leads to such results as

$$\text{'many'} + 1 = \text{'many'}, \quad \text{'many'} + 2 = \text{'many'},$$

which are very familiar to us in the context of views on infinity used in the traditional calculus

$$\infty + 1 = \infty, \quad \infty + 2 = \infty.$$

This observation leads us to the following idea: Probably our difficulty in working with infinity is not connected to the nature of infinity but is a result of inadequate numeral systems used to express infinite numbers.

Thus, it is proposed to introduce a new numeral system having a possibility to express finite, infinite, and infinitesimal numbers in a unique framework and to execute arithmetical operations with all of them. An infinite unit for measuring infinite sets is used as the radix of the new positional numeral system. It is necessary to emphasize that the new approach is not a contraposition to the ideas of Cantor and Robinson. In contrast, it is introduced as an applied evolution of their ideas. The problem of infinity is considered from the point of view of applied Mathematics and theory and practice of computation. The following methodological consideration should be also mentioned.

Note that foundations of the Set Theory dealing with infinity have been developed starting from the end of the XIX-th century until more or less the first decades of the XX-th century. Foundations of the classical Analysis dealing both with infinity and infinitesimal quantities have been developed even earlier, more than 200 years ago, with the goal to develop mathematical tools allowing one to solve problems arising in the real world in that time. As a result, they reflect ideas that people had about Physics more than 200 years ago. Thus, these parts of Mathematics do not include numerous achievements of Physics of the XX-th century.

Even the brilliant results of Robinson were made in the middle of the XX-th century and have been also directed to a reformulation of the classical Analysis

(i.e., Analysis created two hundred years before Robinson) in terms of infinitesimals and not to the creation of a new kind of Analysis that would incorporate new achievements of Physics.

The point of view on infinite and infinitesimal quantities presented in this lecture uses strongly two methodological ideas borrowed from the modern Physics: relativity and interrelations holding between the object of an observation and the tool used for this observation. The latter is directly related to connections between numeral systems used to describe mathematical objects and the objects themselves.

Numerals that we use to write down numbers, functions, etc. are among our tools of investigation and, as a result, they strongly influence our capabilities to study mathematical objects. Moreover, we are able to write down and to study only those numbers that are expressible by numeral systems we know.

The new methodology and the corresponding language allow one to look at problems related to infinity with a higher precision with respect to traditional numeral systems. In [14], infinite processes and Turing machines have been studied using the new approach. In that paper, a deep investigation is performed on the interrelations between mechanical computations and their mathematical descriptions emerging when a human (the researcher) starts to describe a Turing machine (the object of the study) by different mathematical languages (the instruments of investigation). Together with traditional mathematical languages using such concepts as ‘enumerable sets’ and ‘continuum’ the new computational methodology allowing one to measure the number of elements of different infinite sets is used. It is shown how mathematical languages used to describe the machines limit our possibilities to observe them. This analysis is done with respect to deterministic and non-deterministic Turing machines.

Acknowledgments. This research was partially supported by the Russian Federal Program “Scientists and Educators in Russia of Innovations”, contract number 02.740.11.5018.

References

1. Cantor, G.: Contributions to the Founding of the Theory of Transfinite Numbers. Dover Publications, New York (1955)
2. Conway, J.H., Guy, R.K.: The Book of Numbers. Springer, New York (1996)
3. Gordon, P.: Numerical Cognition without Words: Evidence from Amazonia. *Science* 306, 496–499 (2004)
4. Mayberry, J.P.: The Foundations of Mathematics in the Theory of Sets. Cambridge Univ. Press, Cambridge (2001)
5. Robinson, A.: Non-Standard Analysis. Princeton Univ. Press, Princeton (1996)
6. Sergeyev, Y.D.: Arithmetic of Infinity. Edizioni Orizzonti Meridionali, CS (2003)
7. Sergeyev, Y.D.: Blinking Fractals and Their Quantitative Analysis Using Infinite and Infinitesimal Numbers. *Chaos, Solitons & Fractals* 33(1), 50–75 (2007)
8. Sergeyev, Y.D.: A New Applied Approach for Executing Computations with Infinite and Infinitesimal Quantities. *Informatica* 19(4), 567–596 (2008)

9. Sergeev, Y.D.: Evaluating the Exact Infinitesimal Values of Area of Sierpinski's Carpet and Volume of Menger's Sponge. *Chaos, Solitons & Fractals* 42(5), 3042–3046 (2009)
10. Sergeev, Y.D.: Numerical Point of View on Calculus for Functions Assuming Finite, Infinite, and Infinitesimal Values Over Finite, Infinite, and Infinitesimal Domains. *Nonlinear Analysis Series A: Theory, Methods & Applications* 17(12), e1688–e1707 (2009)
11. Sergeev, Y.D.: Numerical Computations and Mathematical Modelling with Infinite and Infinitesimal Numbers. *J. Applied Mathematics & Computing* 29, 177–195 (2009)
12. Sergeev, Y.D.: Computer System for Storing Infinite, Infinitesimal, and Finite Quantities and Executing Arithmetical Operations with Them. EU patent 1728149 (2009)
13. Sergeev, Y.D.: Counting Systems and the First Hilbert Problem. *Nonlinear Analysis Series A: Theory, Methods & Applications* 72(3-4), 1701–1708 (2010)
14. Sergeev, Y.D., Garro, A.: Observability of Turing Machines: A Refinement of the Theory of Computation. *Informatica* (2010) (in press)
15. The Infinity Computer web page, <http://www.theinfinitycomputer.com>

Molecular Implementations of Cellular Automata

Satyajit Sahu¹, Hiroshi Oono², Subrata Ghosh¹, Anirban Bandyopadhyay^{1,*},
Daisuke Fujita¹, Ferdinand Peper³, Tejiro Isokawa², and Ranjit Pati⁴

¹ Advanced Nano Characterization Center, National Institute for Materials Science,
1-2-1 Sengen, Tsukuba, Ibaraki, 305-0047 Japan

anirban.bandyopadhyay@nims.go.jp, anirban.bandyo@gmail.com

² Division of Computer Engineering, University of Hyogo, Shosha 2167, Himeji,
671-2201 Japan

³ Nano ICT Group, National Institute of Information and Communications
Technology, 588-2 Iwaoka, Nishi-ku, Kobe, 651-2492 Japan

⁴ Department of Physics, Michigan Technological University, Houghton,
Michigan, 49931 USA

Abstract. Cellular Automata (CA) have a long history as computation models, but only in the last few years have serious attempts started to implement them in terms of molecules. Such nano-technological innovations promise very cost-effective fabrication because of the regular structure of CA, which allows assembly through molecular self-organization. The small sizes of molecules combined with their availability in Avogadro-scale numbers promises a huge computational power, in which the massive parallelism inherent in CA can be effectively exploited. This paper discusses critical background aspects of our recent results on the implementation of a CA by a molecular assembly (Bandyopadhyay *et al.*, Nature Physics 2010).

1 Introduction

From its inception in the 1950's by von Neumann, CA have attracted interest from researchers in a wide range of fields. The development of the logical base of biological self-reproduction, the *raison d'être* of CA, was soon superseded by the realization of the computing abilities of CA. Over the years this has prompted several attempts towards physical implementations of CA. Prominent in those efforts was the Cellular Automaton Machine (CAM) by Toffoli and colleagues [14] in the 1980's. In the same decade, the chemist Forrest L. Carter conducted research on implementing CA by molecules, in which the exchange of chemical bonds could be used to process information [5]. This endeavor created heated debate at the time, probably due to the yet insufficient levels of technology that could support its realization. The 1990's saw the inception of Quantum-Dot CA and their physical implementations, whereby cells contain four or five dots on

* Corresponding author.

which electrons can reside in certain configurations [12]. In 2002, Heinrich *et al.* [8] reported a molecular cellular automaton in which CO molecules arranged on a copper surface are able to conduct simple logical operations, be it extremely slow.

The distributed nature of CA has prompted comparisons with neural architecture based computers. An important issue in this context is whether different hardware elements can reach a collective decision in an instantaneous way—a mode of operation that is impossible for traditional models [10]. A possible path to such a design is by connecting one device with many others radially through a wireless connection. If thousands of such devices operate synchronously in a massive parallel way, then the resulting processing would enable us to mimic the way a natural phenomenon evolves in reality, revealing unknown features beyond its well-established mechanism. These computers may evolve unique solutions from its astronomical set of choices using configurations never seen before. Recently, proposals for such unconventional computation have attracted great interest [1]; however, they face copious challenges in practical realizations. In these methods, a problem is represented as a logic pattern in the distributed cells, which spontaneously evolves to a distinct logical output pattern as an explicit solution of the problem. Self-evolving logic patterns following particular rules for birth and death of cell states are usually referred to as artificial life, but most of such models lack a physical realization and thus remain limited to theory [7]. In most of the proposals thus far, diffusion or collision of particles/pixels are at the basis of the formation of these patterns; therefore they do not provide atomic scale control [9]. A better degree of control could be achieved using arrayed quantum dots or molecules. However, most such efforts are directed only toward the realization of logic gates and toward proving universal computing abilities [11], rather than focusing on pattern-based computing, which would more effectively be able to exploit the inherent massive parallelism. Our previous model that allows 16-bit parallel processing in molecules [3] is a significant advance in this respect, since it uses of wireless communication to enable one cell to talk to many others at a time, unlike in CMOS-based CA architectures [6].

2 Basic Cellular Automaton Model

By self-assembly, 2,3-Dichloro-5,6-dicyano-1,4-benzoquinone (DDQ) molecules form a honey-comb structure via weak interactions on an atomic flat Au (111) surface, as shown in the Scanning Tunneling Microscope (STM) image in Fig. 1(A) [4]. Each DDQ molecule connects to six neighboring molecules via atomic contacts over a distance of approximately 2 Å. The resulting monolayer represents a hexagonal cellular automaton (HCA), wherein each molecule represents a cell, which communicates with its six neighbors simultaneously and which changes its state following particular rules. The STM image shows an interference pattern of local electron density waves, which suggests that all the molecules are quantum mechanically coupled. The cells have four possible states, which can be observed through differences in contrast of its tunneling-current image. Therefore, we represent a monolayer by a 2-dimensional hexagonal matrix of the four

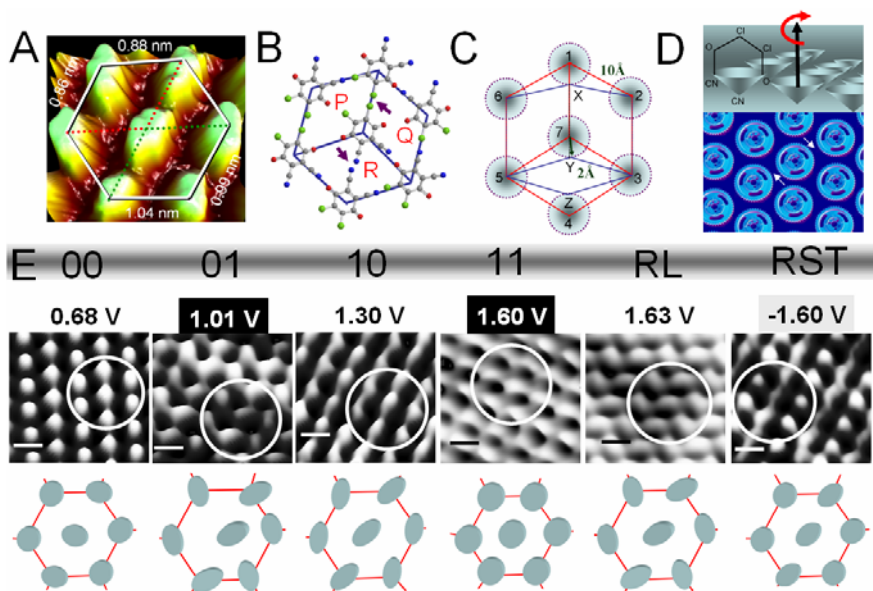


Fig. 1. (A) A 3-dimensional plot of a DDQ hexagon. (B) Structural equivalent of the STM image of the DDQ hexagon; the color codes to identify the atoms are: green (chlorine), red (oxygen), blue (nitrogen), and white (carbon). The blue hexagon is divided into three parts P, Q and R. Arrows denote Cl-Cl and N-N channels. (C) Red hexagon is the ideal one; when the central DDQ is shifted to Y, the upper molecule 1 may or may not shift to the location X, but a distorted blue hexagon is created. Shaded black region is the range where the DDQ can change its position. A larger hexagon has a 1.08 nm side-arm and the smallest hexagon arm observed is 0.8 nm. (D-top) Schematic presentation of energy minimization computed by DFT demonstrating a cone-shaped rotational path of N-O axis of a DDQ in a monolayer. (D-bottom) 2-dimensional array of real gears rotating randomly on a surface. (E) Six columns showing 00, 01, 10, 11, RL (relaxation), and RST (reset) events respectively. There are three rows: in the first row the scan biases are given, the second shows the corresponding STM images with heights varying between 0.10 nm and 0.38 nm, and the third row shows schematics of hexagons revealing the origins of the STM patterns. The scale bar is 1 nm. All images are scanned at respective biases and at 0.05 nA tip current. This schematic is generated by comparing the simulated local densities of states of equivalent molecular structures with the STM images.

numbers 0, 1, 2, 3. These four states correspond with different distributions of electrons on the binding sites of a molecule. States on the surface of the matrix change according to transition rules that originate from the weak interactions. Observation of the changes in the matrix over time gives important information about the rules, which are so versatile that the evolution of a pattern could be manipulated significantly by tuning the initial matrix. In a simple logic pattern, for example, we have encoded basic functions of natural phenomena like heat flow or evolution of cancer cells, and the surface spontaneously created the

solution as a pattern as if it understood the equation assigned to the phenomena in the established theories.

Because of the DDQ's prolate shape (i.e. like a rugby ball pointing up), the central molecule of a basic seven-DDQ assembly shifts with respect to its origin in the course of a rotation. Such a rotation (and translation) of the molecule is triggered by the STM tip during the exchange and tunneling of electrons. Considering all possible origin-shifts due to the reconfiguration of atomic contacts, we found that the DDQ centers trace circular paths. This particular rotation creates unique iso-potential channels connecting atomic contacts between DDQ molecules, which extend throughout the monolayer (Fig. 1(C)). Physically, this corresponds to the creation of distinctly interconnected iso-potential channels traversing between the DDQ molecules in a complicated network 4.

Numerical simulations have been conducted on smaller subsets of the system to better understand the interactions involved. Four molecular conformers of DDQ similar to DRQ 2 were energy-minimized using Density Functional Theory (DFT) computation in a Gaussian 03 platform, using a 6-311G** basis set. At the B3LYP level, states 0 and 2 were obtained by varying the constraint between force and energy. The structure was placed between two gold electrodes composed of a 3×3 matrix of two layers. The gold surface facing the molecule was in a 111-configuration. Self-consistent DFT computation was carried out using Local Density Approximation (LDA), using a numerical basis set, and a strict convergence criterion was employed to a resolution of 0.0001.

It was found that a nitrogen atom of the DDQ that is closer to the Au surface shares its lone pair electron with the s-orbitals of gold atoms, and one of the oxygen atoms forms another bond with the neighboring gold atom. Our calculations further suggest that the molecule essentially survives its boat shape (bent along the O-O axis) even after adsorption with both oxygen atoms pointing towards the gold surface, and the molecule is tilted along the nitrogen side. Periodic density functional theory within the local density functional approach is used to determine the isolated DDQ structure on the Au (111) surface. The super cell structure is constructed from 74 atoms: 60 Au atoms and 14 atoms in a single molecule. From the two layers of Au atoms used in the supercell, the layer that is nearer to the molecule is allowed to relax along with the molecule during structural optimization; the layer that is away from the molecule is kept fixed at the bulk position. The computation is carried out using the VASP code (*Vienna Ab-initio Simulation Package*, which simulates atomic-scale properties of systems) that uses a plane wave basis set and an ultra-soft pseudo-potential to describe the valence-core interaction. During the optimization we used a 111k-point mesh within the Monkhorst-Pack scheme to sample the Brillouin zone. A minimum force criterion of $0.01 \text{ eV}/\text{\AA}$ was used for each individual atom during the structural relaxation. The convergence threshold for energy was taken to be 10^{-6} eV .

To understand the survival of non-zero DDQ states inside a monolayer that generate the changes in the observed STM image, we positioned 6 molecules in a hexagonal pattern around a central DDQ in the same initial configuration as

the monolayer and the seven-molecule system was placed inside a potential box and potential barriers, which atoms can not cross during relaxation. Using initial random orientations of 7 DDQs, we generated optimized orientations, which gave a similar simulated STM image as the experimentally derived one.

3 Some Elementary Cell Patterns

We have created several molecular matrices and applied electronic pulses sequentially in a 13×17 matrix as demonstrated in [4](#). In a continuous scan with a 20 seconds interval at 0.68 V bias, we observed that the targeted input matrix spontaneously changed into a different output matrix at the next scan. Moreover, under similar conditions particular changes occurred repeatedly at different parts of the matrix. To identify the fundamental rules behind this spontaneous reconfiguration, we analyzed the reorganization of logic states locally in a 4×5 matrix based on matrices with the minimal possible sizes.

We have mapped changes in a potential distribution at the atomic contacts, when the cell states were expanding through the monolayer according to the CA's rules. Six side-groups of DDQs, at equilibrium, remain in contact ($< 2 \text{ \AA}$) with the particular side groups of its neighboring DDQs. During structural re-organization, they orient at a maximum of approximately 60° , which makes another side-group restore a one-to-one atomic correspondence. To retain the atomic contacts, the Center of Mass of each prolate-shaped DDQ rotates around a virtual circular periphery inside the matrix. This particular rotation creates unique iso-potential paths along the atomic contacts, and since these contacts determine how electrons flow between molecules, they define the CA's rules. An STM image simulation by a quantum chemistry computation of a molecular structure on a gold (111) surface shows that if any DDQ conformer rotates clockwise inside the matrix around 60° then the Cl and CN groups do not face each other, and eventually for all four states the central DDQ becomes potentially isolated from its neighbors. In this way a non-zero cell state survives when it is created amidst cell states 0. States 3 and state 1 can thus remain localized in this matrix.

A structural analysis of STM images shows that (Fig. [1\(A\)](#)) inside the blue hexagon the pattern is divided into three parts, two of them of equal size denoted as P and R, and the remaining much smaller one denoted as Q. Because of DDQ's prolate shape, the central molecule in the course of a rotation shifts from its origin. As a result, the area distributions of P, Q, and R are redefined (Fig. [1\(B\)](#)). Notably, the formation of an N–N atomic contact separates the DDQs by 1.08 nm (center to center (CC) distance). All contacts formed with N generate a 0.85 nm separation and the rest of the possible atomic contacts generate a CC distance of approximately 0.80 nm. Motions of the DDQs in all hexagons formed by connecting molecules of a monolayer follow a circular domain in which the center of mass (CM) of a DDQ moves during reconfiguration of the atomic contacts by 2 \AA (shaded circular region in Fig. [1\(C\)](#)). As ab initio computation has shown that the nitrogen atom of DDQ is nearer to the surface gold atoms

by 1 Å, as compared to an oxygen atom, the rotational path of CM does not traverse on a planar disk, but rather follows a conical surface (Fig. 1(D-top)). The motion of the molecules inside a monolayer is similar to an array of a large number of gears in a machine, where the gears stabilize themselves only when the teeth of two different gears face each other (Fig. 1(D-bottom)).

The potential distribution for the atomic arrangement of a particular hexagonal architecture is calculated again using DFT to understand the effect of the asymmetric rotations demonstrated above. In contrast to previous simulations, additional changes in the CM position and a lifting up of a DDQ are included in the computation. A planar rotation by approximately 60° and an additional lifting up of a particular region of DDQs by 1 Å during a change in the CM results in an additional separation of 2 Å between the nearest neighbor groups that form a minimal potential contact with the central molecule (Fig. 1(E)). This makes the central DDQ completely isolated potentially from its neighbors.

4 Cell Patterns Used in Simulations of Physical Phenomena

Computation on CA is a local process: the next state of a cell is determined by the interactions between the cell and its nearest neighbors only. Each cell is a finite state machine, and several cells, as well as cell configurations representing classical logic gates, have been built experimentally using quantum dots, single molecules, etc. However, computation in which a decision pattern is generated through the interactions among a large number of cells has not been realized yet, even though theoretical models are in abundance in the literature. If realized, one would obtain a fast evolution toward the solution of a problem, i.e. many cells would be involved simultaneously in the emergence of a solution.

In this way, the essential mechanisms of different natural phenomena can be captured by varying the initial pattern and the proper combination of DDQ conformers inside the sub-patterns. Classically, to solve a differential equation, we consider a generalized solution and then determine the coefficients following rules according to certain assumptions. However, in a pattern-based solution, only the key features of the phenomenon are encoded. The encoded pattern and the CA rules are so correlated that every single step taken by the pattern follows the equation; in particular the end-points of patterns continuously provide solutions of differential equations. Thus, man-made equations are not required in such a CA universe or in nature. This encoding process would be as versatile as our day-to-day computer programming if a distinct pattern could be used as a basic unit that would remain constant during pattern evolution.

Different regions of the bi-layers of a DDQ molecule on an Au (111) surface reorient into distinctly different ordered arrangements as soon as excess electrons are injected to the assembly. Similar to a circuit, the assembly then triggers the excess electrons to propagate following particular rules and finally form a new map of electrons. Manipulating the local architecture, we tune the propagation of electrons as if they diffuse through the surface similar to a heat flow or to

the redistribution inside a circle as if normal cells mutate to cancer cells inside a tissue. Since electron propagation is visible in the tunneling current image, we represent the complete event as interplay of balls with four colors representing the four cell states. Thus, electron propagation rules encode cellular automata (CA) rules, and the monolayer will act like a CA grid. The electrons either enter a molecule (birth of a CA cell), remain there (survive) or leave (death), similar to artificial life simulations.

For each particular input pattern, we observed spontaneous flipping of weak bonds between molecules, building a new communication circuit by creating and destroying several optional paths connecting hundreds of cells. Building a new circuit turns it creative, simply because the total number of circuits that could be generated for a particular kind of problem is nearly infinite. Unlike supercomputers, the hardware itself talks to multiple cells at a time, corrects error in the process, and thus it exhibits a form of intelligence. During execution, exchange of an electron or a few kilo-calories of energy enables wireless computation with minimum external power supply. While a Pentium IV processor with a device density of $< 10^9/\text{cm}^2$ dissipates approximately $100 \text{ W}/\text{cm}^2$, the DDQ-HCA with a device density of $10^{14}/\text{cm}^2$ dissipates approximately $1 \text{ W}/\text{cm}^2$; thus heat generation is significantly minimized. In principle, a solution is generated collectively, so even if some cells stop working suddenly, the entire computation does not collapse, but rather the system reaches a solution.

5 Discussion and Conclusions

DDQ molecules tend to act in coordination with many other molecules in large areas, whereby the electrical charge density in an area makes the molecules incline in similar ways. The area-wise inclination of molecules results in structures called *networks* (or *circuits*)—a name adopted because of the tendency of electrical charges to travel from molecule to molecule via the geometrically shortest path between molecules. Areas of molecules are distinguished by their type, several of which are shown in [4]. The type of a network to which a molecule belongs is not to be confused with the state of the molecule, which is only determined by the distribution of electrical charge on the molecule. A network type has no counterpart in traditional models of CA. It does, however, influence the dominance of transition rules, which means that the probability that a rule is applied to a certain cell increases or decreases depending on the network type, as demonstrated in [4]. This characteristic of DDQ-based CA makes them very useful from a computation point of view, since it gives control over the type of rules that can be applied on molecules via a single parameter that is the charge density in localized areas. It allows the complicated functionality of the molecular layers to be represented by relatively simple molecules. The formation of networks on the molecular layer appears to be simultaneous: molecules in an area tend to change their network type as a massively parallel operation, affecting all of them in the area. This brings us to the topic of timing.

Timing in CA was originally thought to be synchronous by von Neumann, meaning that all cells will be updated at the same time in successive discrete

time steps. This mode of updating according to a central clock signal has set the tone for most CA research to date, but there have been efforts by a minority of CA researchers to follow a more flexible update mode. Called asynchronous updating, this timing mode is everything what synchronous updating isn't: the selection of cells for update and the update times are determined in a random fashion, thus abandoning the strict requirement of simultaneous updating of cells. Asynchronous updating has found its way in CA simulations of processes in nature for the obvious reason that those processes are usually not (micro-)managed by a central clock signal. It has also found application for CA intended for models of computation on nanometer scales [13], for similar reasons: it is likely easier to let molecules process in their own natural way than trying to synchronize them.

So, do the DDQ molecules in our model behave in an asynchronous way? Though the lack of a central clock may suggest an affirmation to this question, surprisingly the answer is not as straightforward. Since networks of molecules change their type simultaneously, as pointed out above, there is an element of synchronicity involved, but network types tend to remain within areas of similar charge densities. The change of network types thus appears as a mixture of synchronous and asynchronous timing. For the update of cell states, the picture looks different, however: this appears to be a purely asynchronous affair. That does not exclude determinism of behavior: like the asynchronous CA in [13], the DDQ-based CA are able to conduct certain deterministic operations.

Nature exhibits sophisticated collective information processing capabilities that show similarities to our brain, the reproduction of multi-cellular organisms, and so on. In this context, global co-ordination emerges from the decentralized communication between simple components. This particular feature has important advantages over man-made supercomputers where a central unit explicitly controls all computation processes. The first advantage is that all functional parts no longer need to be connected to the central control via physical wiring in order to increase the speed (approximately 10 km wiring/cm² area of an integrated chip). The second advantage is that the loss of connection with a central control unit will not jeopardize the entire system, thus making it more robust. The third advantage is that allocated resources are equally divided. In the past such wireless and powerless computation has been proposed in theoretical models cellular automata.

Cellular automation may be used to model artificial life that follows defining characteristics of living systems. We have demonstrated the existence of such artificial life forms as a logic pattern that evolves in hundreds of molecules in a molecular layer. The layer consists of DDQ molecules that reversibly switch between four states. A molecule may change states of its six (or, depending on the local network configuration: four) neighbors at a time following particular transition rules for the birth, survival and death of cells and thus create a new electron transport circuit for a new problem. The way patterns change over time is thus strongly dependent on their arrangement on the molecular layer, and we have especially observed this for two types of patterns, i.e. linear and

circular patterns. For the first, the pattern changes over time as if electrons diffuse throughout the surface, and for the second the pattern changes as if normal tissue cells mutate continuously to give rise to cancer cells [4].

Individual molecules are wired quantum-mechanically to each other, and communication between them enables wireless information transport without any power supply required. Since CA can accurately model numerous real-world phenomena and systems ranging from genetics to economics, the physical realization of such CA would lead to a better understanding of the world around us. Even though we have reproduced only a few kinds of supercomputing processes on the molecular layer, other fundamental properties like those found in the central nervous system such as adaptation, plasticity, and self-organization, might be realized in the near future on molecular layers.

Acknowledgments. Authors acknowledge JSPS Grants in Aid for Young Scientists (A) for 2009-2011, Grant number 21681015 (Govt of Japan). R.P. acknowledges National Science Foundation (NSF) Award number ECCS-0643420.

References

1. Adamatzky, A., Teuscher, C. (eds.): *From Utopian to Genuine Unconventional Computers*. Uniliver Press, Frome (2006)
2. Bandyopadhyay, A., Miki, K., Wakayama, Y.: Writing and erasing information in multilevel logic systems of a single molecule using scanning tunneling microscope (STM). *Appl. Phys. Lett.* 89(24), 243507 (2006)
3. Bandyopadhyay, A., Acharya, S.: A 16-bit parallel processing in a molecular assembly. *Proc. Natl. Acad. Sci.* 105(10), 3668–3672 (2008)
4. Bandyopadhyay, A., Pati, R., Sahu, S., Peper, F., Fujita, D.: Massively parallel computing on an organic molecular layer. *Nature Physics* 6(5), 369–375 (2010)
5. Carter, F.L.: The molecular device computer: point of departure for large scale cellular automata. *Physica D* 10(1-2), 175–194 (1984)
6. Durbeck, L.J.K., Macias, N.J.: The Cell Matrix: an Architecture for Nanocomputing. *Nanotechnology* 12(3), 217–230 (2001)
7. Gaylord, R.J., Nishidate, K.: *Modeling Nature*. Springer, Santa Clara (1996)
8. Heinrich, A.J., Lutz, C.P., Gupta, J.A., Eigler, D.M.: Molecule Cascades. *Science* 298(5597), 1381–1387 (2002)
9. Hjelmfelt, A., Weinberger, E.D., Ross, J.: Chemical implementation of neural networks and Turing machines. *Proc. Natl. Acad. Sci.* 8, 10983–10987 (1991)
10. Higuchi, T., Murakawa, M., Iwata, M., Kajitani, I., Liu, W., Salami, M.: Evolvable hardware at function level. In: *Proc. IEEE Conf. Evolutionary Computation (ICEC)*, pp. 187–192 (1997)
11. Imre, A., Csaba, G., Ji, L., Orlov, A., Bernstein, G.H., Porod, W.: Majority Logic gate for magnetic Quantum-dot Cellular Automata. *Science* 311(5758), 205–208 (2006)
12. Lent, C.S., Tougaw, P.S., Porod, W., Bernstein, G.H.: Quantum Cellular Automata. *Nanotechnology* 4(1), 49–57 (1993)

13. Peper, F., Lee, J., Adachi, S., Mashiko, S.: Laying Out Circuits on Asynchronous Cellular Arrays: a Step towards Feasible Nanocomputers? *Nanotechnology* 14(4), 469–485 (2003)
14. Toffoli, T.: CAM: A High-Performance Cellular Automaton Machine. *Physica D* 10(1-2), 195–205 (1984)

Achieving Universal Computations on One-Dimensional Cellular Automata

Jean-Baptiste Yunes

Laboratoire d'Informatique et Algorithmique, Fondements et Applications (LIAFA),
Université Paris Diderot & CNRS, Case 7014, 75205 Paris Cedex 13
`Jean-Baptiste.Yunes@liafa.jussieu.fr`

Abstract. We show how natural universal computations can be achieved on one-dimensional cellular automata. That model of computation is obviously Turing complete but how can we effectively program any given computation is far less known. In this paper we are interested in intrinsic universality; we want a CA in which any other CA can be represented and simulated with no intermediate coding relevant to another computation model. The first step is to abstract from the space-time diagram in favor of a more essential dependency graph. Then such dependency graph can be projected on grids. This work shows that grids put forward causality in place of space-time contingencies.

1 Introduction

In this paper, we first exhibit very simple computations on one-dimensional cellular automata, CA. We observe how usual algorithms are fundamentally parallel and can be easily implemented on CA. By observing how information flows during the process we are able to explain how the task of programming on such a model can be done by abstracting the contingencies of the machine — in particular space and time — to a more general concept of causality. This is how the space-time diagram is replaced by a grid. A grid is simply the universal basic dependency graph of any CA and can then realize any CA computation.

A simulation is first achieved by the dynamical construction of a grid into the space-time diagram of a CA viewed as the hardware, and by the projection of the space-time diagram of a CA to be simulated into the grid.

Universality is then built by injecting the coding of the transition rules of a CA into the whole process.

2 Examples of Computations

2.1 Clocking and Subclocking

That first example is probably of no practical use, but will serve us as an illustration of a very simple but fundamental principle for CA programming.

The construction produces a family of clocks all derived from a main clock. Each successive subclock ticks one tick over two of the previous clock, so that if

the main clock is periodic with period p , on cell i the period will be $2^i p$. The behavior is quite simple and exactly reflects what one would have done in real life: if someone ticks you, you just have to wait two ticks before ticking yourself as in Fig. 1. In this figure, ticks are represented by black squares.

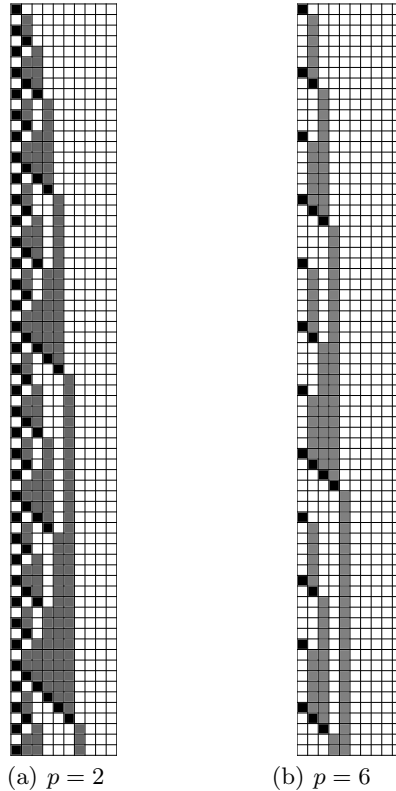


Fig. 1. Two clocks of period p and their subdivisions. (Time going downward.)

Let us emphasize the nature of the construction. In fact, whatever the first cell produces (regular clocking or not, even if not algorithmically produced), the algorithm produces for each cell a subclock (one tick over two) of the one produced by the previous cell. We distinguish here two important things:

- the initial process (what is produced on the first cell);
- the iterative process (what is produced from the flow of information transmitted by the previous process). This locality feature is the crux of programming, and the source of modularity.

Processes are standing against one another: the previous providing inputs for the next. This geometrical construction is the main idea which underlies the

concept of grids and leads to a very natural implementation of computation composition.

2.2 From Clocks to Firing Squad Synchronization Solutions

In the previous example the construction tightly intermixes the machine and the algorithm realized by the machine. It may seem that the construction is tricky whereas explanations are quite simple. In fact, what really matters is the logic of the construction. The hardware contingencies may be a source of confusion if the logic has not been properly clarified. We now give a much more interesting example.

The problem can be stated as the following:

Whatever be the length of the line, from an “empty” initial configuration — empty except for the first cell — the transition rules should make all cells enter simultaneously in the same state for the first time.

Among the numerous solutions, we focus on one that has interesting properties for our discussion. That solution is due to Mazoyer (see [7]) and is illustrated in Fig. 2.

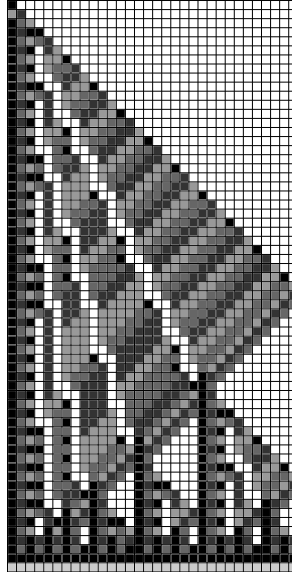


Fig. 2. Mazoyer’s solution on a line of length 32. (Time going downward.)

What can be observed is that the recursive process uses some kind of clock division process. One can see a kind of period made of three states that fills

the first main triangle, its first subdivision in the second triangle which stands against the first (the frontier is represented by the blank signal), etc.

Except for technical details, one can see that the algorithm in this solution is based on a kind of clock division algorithm similar to the previous example. The main difference is that no clock is located on a single cell anymore: each clock has to be realized by some computation that takes place in a dedicated part of the space-time diagram. Each clock runs in a triangle delimited by blank signals. For each clock what exactly happens inside the dedicated zone of other clocks is of no importance, only the communication that takes place at blank signals is meaningful. If it seems odd to the reader that the quiescent state has been used as a signal, this is just a trick used by Mazoyer to optimize the total number of states of his solution. The reader may imagine that this could be another dedicated state. That “frontier” signal is the border of triangles one standing against another.

So what makes the real difference in both examples is that in the second one the main direction of any clock (see Sablik [III]) is not vertical but a line with some slope. Further it moves in the space during the computation and uses more and more cells as time grows. These moves constraint the moves of the next subclock, thus producing related speeds of clock moves (they are related in the same way as are their periods).

2.3 Addition and Multiplication

In the previous examples, we started from implemented computations and abstracted them to kinds of blocks of computations together with their geometrical organization. Now we proceed in the converse way. We start with the two elementary arithmetical operations, plus and times, and use them to illustrate how we can organize computations in the space-time diagram of CA.

$\begin{array}{r} 148 \\ + 274 \\ \hline 422 \end{array}$	$\begin{array}{r} 148 \\ \times 274 \\ \hline 592 \\ 1036\ . \\ 296\ \cdot\cdot \\ \hline 40552 \end{array}$	$\begin{array}{l} =148 \times 4 \\ =148 \times 70 \\ =148 \times 200 \end{array}$
(a) Addition	(b) Multiplication	

Fig. 3. The usual addition and multiplication algorithms

What should be remarked is that in these two algorithms results are produced, on each line, digit by digit and from right to left. This is due to propagation of carries which induces dependencies between digits as illustrated in Fig. 4.

A simple rotation of those graphs, as in Fig. 5, shows how these graphs are just space-time diagrams of trellis automata.

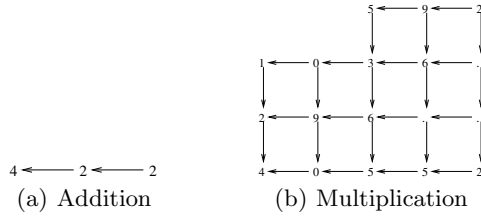


Fig. 4. Dependency graphs as treillis automata

Observe that we removed many unnecessary technical details of the algorithms like carry management, value bounding, etc. For more information about this, we refer to [1] for the multiplication and to [8] for these computations in the context of grids.

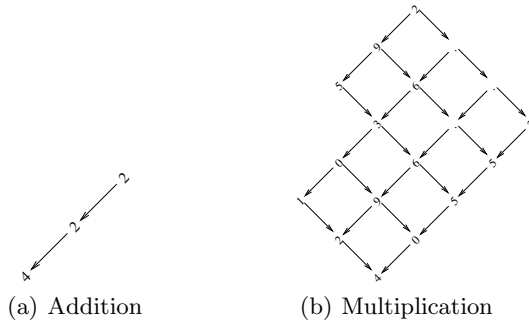


Fig. 5. Dependency graphs

One can remark that the dependency graphs we obtain can be projected in the space-time diagram of any CA. There were two approaches to this question which are due to Cole (see [3]) and Choffrut & Čulik (see [2]). Whatever approach is taken, the result is the same and the general idea is the following (from section 5.1 in [8]):

It is sufficient to have two independent communication channels to be able to simulate any one-dimensional CA whatever be its neighborhood.

Every dependency graph of a CA computation is the Cayley graph of a monoid with two generators and every such Cayley graph can be embedded in a dependency graph. Thus, in a fundamental sense, this means that what really matters is the dependency graph, not the way it is embedded into a CA machine. The programmer must focus on the underlying dependency graph of the computation to be implemented. He/she must formalize it, and the compiler will embed it as needed.

Observe that in a dependency graph there is no concept of space or time.

3 Computing on Grids

We previously stressed that what matters is the dependency graph: the implementation should not interfere with the logic. Of course to compute something, one needs to exhibit an adequate embedding of the dependency graph into the space-time diagram of the machine. So a question arises: how can we embed it? A first answer is provided by Cole (see [3]) and Choffrut & Čulik (see [2]) embeddings. These embeddings are just as regular as the Cayley graph is. We want to overcome any physical contingency related to such a regularity: it should be an abstract regularity rather than a physical one given by neighborhoods $\{-1, +1\}$, $\{0, +1\}$ or $\{-1, 0\}$ as in Cole and Choffrut & Čulik. This leads to the concept of grids: roughly, a grid is any representation — possibly quite irregular — of a treillis. The sole constraint is that any causality in the representation has to be a causality in the implementing machine (*i.e.* respects the dependency cones). We refer the reader to [10] for question on complexity relative to neighborhood equivalence.

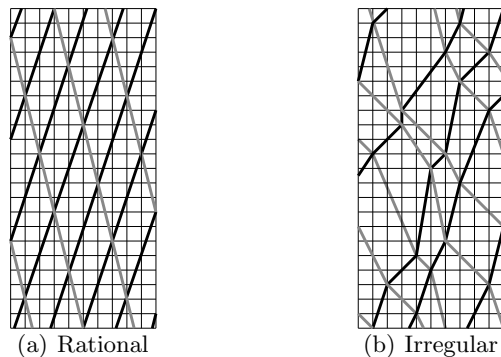


Fig. 6. Different embeddings of a treillis in classical CA. (Time going downward.)

Given a CA with neighborhood $\{-1, 0, +1\}$ one can construct in this CA many different representations of the dependency graph of a given computation. What does mean construct? A possible informal answer is to draw such a graph in the space-time diagram of the given CA. The generators of the monoid of the dependency graph can be represented by CA signals (see [12]) that meet. Any meeting corresponds to an element of the monoid. Though it is easy to imagine a very regular embedding as shown in Fig. 6(a) (this is called a rational grid since slopes of signals are rationals and constant), it is possible to use more general embeddings as illustrated in Fig. 6(b). Note that any usable embedding must be constructible by a CA with neighborhood $\{-1, 0, +1\}$ from a finite initial configuration.

3.1 Simulation

Suppose that on a given elementary initial configuration (a unique cell not quiescent), the CA \mathcal{G} is able to construct a grid as defined in the previous section. Example of such a grid constructor are shown in figures 7(a) and 7(b). Given another CA \mathcal{S} to be simulated on its initial configuration $I_{\mathcal{S}}$ (an example is shown in Fig. 7(c)), it is possible to construct a CA Sim , that is able to embed the space-time diagram of \mathcal{S} on $I_{\mathcal{S}}$ in the grid constructed by \mathcal{G} .

Fig. 8 shows how the whole process is done. Fig. 8(a) shows a rational embedding of the space-time diagram of CA of Fig. 7(c). Things are pretty clear when the grid is rational: it is easy to see that each column embeds the trace of the simulated CA. When the grid is irregular, as show in Fig. 8(b), it is much more difficult to see the original space-time diagram, but it is sufficient to remark that one can simply bend the rational embedding as needed.

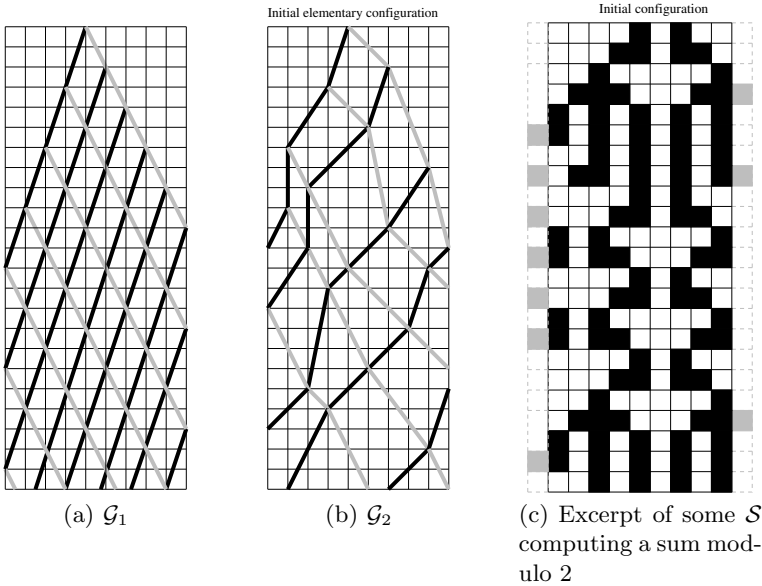


Fig. 7. Elements of a simulation. (Time going downward.)

In the illustrations, we omit many technical details concerning the relative positioning of the two initial configurations of \mathcal{G} and \mathcal{S} , we refer the interested reader to [8]. Let us stress that this construction is uniform.

3.2 Universality

To achieve universality it is necessary to be able to inject a coding of the transition rules of the machine to be simulated into the simulating machine so as to be able to execute any needed transition rule at each node of the grid.

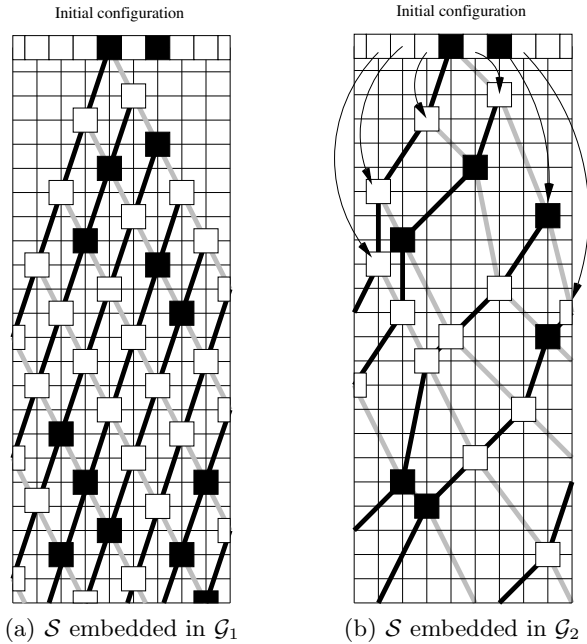


Fig. 8. Simulation. (Time going downward.)

In previous embeddings, meeting points correspond to elementary cells of the underlying machine. This can be generalized: a meeting point can be a piece of the underlying space-time diagram. The idea is that in this piece of the space-time diagram any complex computation can be done. In particular, a computation that produces the result of the local transition function of the simulated CA. Such a computation can be, for example, the one by Martin (see [6]) or by Ollinger (see [9]).

The injection of the transition rules to the nodes of the grid, can be made by a very similar construction of the injection of the initial configuration of the CA to be simulated.

4 Towards Grids

Let us recall the notions of intrinsic universality. The one by Durand & Róka, see [5], requires that a single step of an automaton is simulated by a fixed number of steps of another automaton (up to some coding of configurations). The second one by Ollinger, see [9], introduces a grouping operation. These definitions preserve the strong regular nature of CA (either temporal or spatio-temporal regularity). In our work such a regularity can be kept or removed as wanted. In a grid, computations can be made asynchronously. This gives a powerful flexibility to implement interesting operations with grids.

For example, a composition is easy to obtain. It is sufficient to distort two computations such that outputs of one of the computations correspond to inputs of the other. Moreover, as any trellis distortion is a suitable grid (remind that this means that the distortion respects causality of the machine), one can even imagine dynamic distortions, *i.e.* distortions driven by another computation or by the computation itself. A simple example of the latter could be obtained when output bits of a computation are located on some non regular frontier. Then, any computation can stand against it, thereof uses an irregular border as its inputs and thus computes on a dynamic grid. Thus, when programming some function, it is no more necessary to output bits of results along some regular frontier. Whatever way the output is generated, the computation can be injected on a grid as it is, and even combined into a composition. In some way, this leads to computations that synthesize their circuits during their runs.

The general schema does not require any regularity. What is essential is to focus on causality. Afterall, this is the only unavoidable feature in computing: if you want a machine to execute a program, the causalities induced by the machine must be compatible with the causalities expressed by the program.

Acknowledgments. The author would like to thank Pr. Jacques Mazoyer for all his helpful and constructive discussions on the topic and also Pr. Serge Grigorieff who generously reviewed the draft of this paper.

References

1. Atrubin, A.J.: A one-dimensional real-time iterative multiplier. *IEEE Transactions on Electronic Computers* 14, 394–399 (1965)
2. Choffrut, C., Čulik II, K.: On real-time cellular automata and trellis automata. *Acta Informatica* 21, 393–407 (1984)
3. Cole, S.N.: Real-time computation by n-dimensional iterative arrays of finite-states machines. *IEEE Transactions on Computers* C-18/4, 349–365 (1969)
4. Delorme, M., Mazoyer, J.: Cellular Automata: A Parallel Model. In: *Mathematics and Its Applications*, vol. 460, ISBN: 0-7923-5493-1. Kluwer Academic Publishers, Dordrecht (1999)
5. Durand, B., Róka, Z.: *The Game of Life: universality revisited*. In: *Cellular automata*, Saissac, pp. 51–74. Kluwer Acad. Publ., Dordrecht (1996)
6. Martin, B.: *A universal cellular automaton in quasi-linear time and its S-m-n form*. *Theoretical Computer Science* 123, 99–237 (1994)
7. Mazoyer, J.: A six states minimal time solution to the firing squad synchronization problem. *Theoretical Computer Science* 50, 183–328 (1987)
8. Mazoyer, J., Yunès, J.-B.: Computations on Cellular Automata. In: Rozenberg, G., Bäck, T., Kok, J. (eds.) *Handbook of Natural Computing*. Springer, Heidelberg (to appear, 2010) ISBN: 978-3-540-92911-6
9. Ollinger, N.: The intrinsic universality problem of one-dimensional cellular automata. In: Alt, H., Habib, M. (eds.) *STACS 2003*. LNCS, vol. 2607, pp. 632–641. Springer, Heidelberg (2003), doi:10.1007/3-540-36494-3
10. Poupet, V.: Cellular Automata: Real-Time Equivalence Between One-Dimensional Neighborhoods. In: Diekert, V., Durand, B. (eds.) *STACS 2005*. LNCS, vol. 3404, pp. 133–144. Springer, Heidelberg (2005)

11. Sablik, M.: Directional dynamic for cellular automata: A sensitivity to initial condition approach. *Theoretical Computer Science* 400(1-3), 1–18 (2008)
12. Mazoyer, J., Terrier, V.: Signals in One-Dimensional Cellular Automata. *Theoretical Computer Science* 217(1), 53–80 (1999), doi:10.1016/S0304-3975(98)00150-9
13. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Inc., Champaign (2002)

Author Index

- Acconci, Vito 334
Alt, Leonardo S. 275
Andreadis, Ioannis Th. 455
Andrés Montoya, J. 153
Appert-Rolland, Cécile 542
Avolio, Maria Vittoria 83
Ayala, Rosa 312
- Bachhar, Tirthankar 46
Baetens, Jan M. 95, 177
Bagnoli, Franco 188
Bandini, Stefania 334, 345, 385
Bandman, Olga 395
Bandyopadhyay, Anirban 650
Bastanfard, Azam 79
Belbasi, Somayyeh 138
Bhaumik, Jaydeb 231
Blecic, Ivan 106
Bonomi, Andrea 334, 345, 385
- Caballero-Gil, Pino 251
Cacciagrano, Diletta 116
Calidonna, Claudia R. 128
Cecchini, Arnaldo 106
Cepolina, Elvezia M. 446
Chakrabarti, Indrajit 231
Chopard, Bastien 163
Colacci, Annamaria 1
Corradini, Flavio 116
Costa, Jose 173
Cottenceau, Guillaume 12
- Daadaa, Yassine 287
Dai, Shiqiang 532
Dalui, Mamata 300
Damiani, Chiara 1
Das, Sourav 241
Das, Sukanta 300
De Baets, Bernard 95, 177
Désérable, Dominique 12
Di Gregorio, Salvatore 83, 128
Di Stefano, Bruno 322, 600
Dong, Liyun 532
- Ebbinghaus, Maximilian 542
Ebrahim Foulaadvand, M. 138
Ediger, Patrick 24
El Yacoubi, Samira 188
Errera, Alessia 83
Espínola, Moisés 312
- Farina, Alessandro 446
Ferreira, Giordano B. 275
Flocchini, Paola 287
Frehmel, Sebastian 35
Fricke, Hartmut 489, 506
Fujita, Daisuke 650
Fúster-Sabater, Amparo 251
- Gao, Zi-You 613
Georgoudas, Ioakeim G. 455
Ghaemi, Mehrdad 142
Ghosh, Soumyabrata 46, 271
Ghosh, Subrata 650
Goldengorin, Boris 149
Gruau, Frédéric 69
- Hoffmann, Rolf 24
Huang, Ding-wei 552, 557
Huang, Wei-neng 557
- Imamura, Takashi 561
Iribarne, Luis 312
Ishii, Hideaki 465
Isokawa, Teijiro 650
Ito, Nobuyasu 570
Iwata, Yoshio 58
- Jia, Bin 613
Jiang, Rui 523
- Kalgin, Konstantin 399
Kamimura, Atsushi 570
Kanai, Masahiro 580
Karmakar, Sandip 261
Kauffman, Stuart A. 1
Kirik, Ekaterina 474
Koltsidas, Georgios 455
Komatsuzaki, Toshihiko 58

- Kretz, Tobias 480, 489, 589
 Krougllov, Dmitriy 474
 Krushinski, Dmitry 149
 Kubo, Keisuke 219
- Lawniczak, Anna T. 322, 600
 Lee, Jia 356
 Leguizamón, Saturnino 312
 Lembo, Paola 439
 Li, Xin-Gang 613
 Lupiano, Valeria 83
 Lygouras, John 373
- Macauley, Matthew 409
 Maignan, Luidnel 69
 Maiti, Nirmalya S. 46, 271
 Makarenko, Alexander 149
 Manenti, Lorenza 439
 Manzoni, Luca 419
 Manzoni, Sara 439
 Martins, Luiz G.A. 275
 Matsumoto, Shigenori 570
 Mauri, Giancarlo 429
 Mazzanti, Paolo 83
 Mejía, Carolina 153
 Menenti, Massimo 312
 Merelli, Emanuela 116
 Miki, Hiroshi 619
 Minemura, Takumi 593
 Minoofam, Seyyed Amir Hadi 79
 Miranker, Willard 173
 Mitra, Indrajit 46
 Morishita, Shin 465
 Mortveit, Henning S. 409
 Mukherjee, Sukanya 300
 Mukhopadhyay, Debdeep 261
 Musienko, Anton 149
- Naddeo, Adele 128
 Naderi, Omid 142
 Naskar, Nazma N. 300
 Nishinari, Katsuhiko 513, 523, 593,
 619, 625
 Nishi, Ryosuke 619, 625
- Ohira, Toru 570
 Ohtsuka, Kazumichi 523
 Oliveira, Gina M.B. 275
 Oono, Hiroshi 650
- Pal Chaudhuri, Parimal 46, 271
 Pati, Ranjit 650
 Peper, Ferdinand 356, 650
 Piwonska, Anna 198
 Portz, Andrea 496
- Razakanirina, Ranaivo Mahaleo 163
 Rechtman, Raúl 188
 Roy Chowdhury, Dipanwita 231,
 241, 261
- Saavedra, Patricia 633
 Sahu, Satyajit 650
 Santen, Ludger 542
 Schadschneider, Andreas 496, 593
 Schultz, Michael 489, 506
 Schulz, Matthias 209
 Seredynski, Franciszek 198
 Sergeev, Yaroslav D. 646
 Serra, Roberto 1
 Seyfried, Armin 496
 Shikdar, Biplab K. 271
 Shimura, Kenichiro 513
 Sikdar, Biplab K. 300
 Sirakoulis, Georgios Ch. 373, 455
 Suma, Yushi 523
- Tanaka, Yuki 513, 523
 Tomoeda, Akiyasu 523, 619, 625
 Trunfio, Giuseppe A. 106, 128
 Tsiftsis, Anastasios 373
 Tuck, David 173
- Umeo, Hiroshi 219
- Valsecchi, Andrea 429
 Vanneschi, Leonardo 429
 Velasco, Rosa María 633
 Villani, Marco 1
 Vizzari, Giuseppe 334, 345, 385
- Woelki, Marko 637
 Wu, Hao 322
- Yanagisawa, Daichi 523, 619
 Yunès, Jean-Baptiste 660
 Yurgel'yan, Tat'yana 474
- Zabihinpour, Zahra 142
 Zaguia, Nejib 287
 Zawidzki, Machi 365
 Zhang, Peng 532