Membrane Computing Models: Implementations

Gexiang Zhang • Mario J. Pérez-Jiménez •
Agustín Riscos-Núñez • Sergey Verlan •
Savas Konur • Thomas Hinze •
Marian Gheorghe

Membrane Computing Models: Implementations

Springer

Gexiang Zhang
School of Control Engineering
Chengdu University of Information Technology
Chengdu, China

Agustín Riscos-Núñez
Department of Computer Science and Artificial Intelligence
University of Seville
Sevilla, Spain

Savas Konur
Department of Computer Science
University of Bradford
Bradford, West Yorkshire, UK

Marian Gheorghe
Department of Computer Science
University of Bradford
Bradford, West Yorkshire, UK

Mario J. Pérez-Jiménez
Department of Computer Science and Artificial Intelligence
University of Seville
Sevilla, Spain

Sergey Verlan
Département Informatique, LACL
Université Paris Est Créteil
Creteil cedex, France

Thomas Hinze
Department of Bioinformatics
Friedrich Schiller University Jena
Jena, Thüringen, Germany

ISBN 978-981-16-1565-8 ISBN 978-981-16-1566-5 (eBook)
https://doi.org/10.1007/978-981-16-1566-5

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2021, corrected publication 2021
This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Dedicated to Academician Gheorghe Păun's 70th Anniversary.

Foreword

An Interlaced Triad: Theory, Applications and Implementations

The present book nicely completes and illustrates the triad mentioned in the title above for the *membrane computing* research area.

Membrane computing (a more suggestive name could be *cellular computing*) is a branch of natural computing aiming to abstract computing models from the structure and the functioning of biological cells, considered alone or cooperating in populations of cells, tissues, and organs, neuronal nets included. The obtained models form a rather diverse-versatile *modeling framework*, characterized by a series of features which are attractive from a theoretical computer science point of view (especially in what concerns imagining new computing devices, involving new ideas concerning the computer architecture, data structures, operations with these structures, ways to organize/control the computations, and so on, with a direct and promising influence on the computing power and on the computational complexity/efficiency), but also very important from the point of view of applications (the models are distributed and parallel, involving discrete mathematics—which is adequate to a large variety of processes, from biology to economics, from linguistics to engineering, where traditional tools based on continuous mathematics, typically, differential equations, cannot be applied). From the applications point of view, the membrane computing models have also other attractive features—I only mention some of them: easy scalability/extensibility, non-linear complex behavior, easy programmability, and understandability/transparency.

In more than two decades since membrane computing research was initiated, a large bibliography was accumulated (see, e.g., the information provided by http://ppage.psystems.eu/ and by http://imcs.org.cn/—the latter one is the web page of the International Membrane Computing Society). As expected, initially mainly theoretical computer science investigations were carried out: variations of models (shortly called *P systems* in the community), trying to capture more and more biological features or motivated by "classic" computer science, results concerning the computing power, in comparison with usual computing hierarchies, of Turing, Chomsky types and related, computational complexity results. The results are

The original version of this book was revised. A correction to this book is available at https://doi.org/10.1007/978-981-16-1566-5_9.

somewhat expected for a natural computing model: Turing universality, reached sometimes with unexpectedly "simple" models; speed-up of computations, until theoretically solving intractable problems (typically, **NP**-complete problems) in feasible time (typically, polynomial); and so on.

Soon (actually, already in the first years of developing this research area) applications were explored, initially in biology—which is rather natural: the model is abstracted from the cell biology and the biologists need new (computational, discrete, understandable) models for investigating the cell. In time, the range of applications got wider and wider, while the applications became more and more sophisticated, hence more relevant for the users.

I recall here only three books (two are collective volumes) reporting applications of P systems, also suggestive in what concerns the areas of applications (rather general/diverse for the first volume: cell biology, computer graphics, cryptography, linguistics, sorting):

Gabriel Ciobanu, Gheorghe Păun, Mario J. Pérez-Jiménez (Eds.), *Applications of Membrane Computing*, Springer-Verlag, 2006.
Pierluigi Frisco, Marian Gheorghe, Mario J. Pérez-Jiménez (Eds.), *Applications of Membrane Computing in Systems and Synthetic Biology*, Springer-Verlag, 2014.
Gexiang Zhang, Mario J. Pérez-Jiménez, Marian Gheorghe, *Real-Life Applications with Membrane Computing*, Springer-Verlag, 2017.

Many other similar volumes can be found in the bibliographies indicated above.

Especially for the applications, but also for theory (towards the application edge, e.g., of computational complexity), of a direct interest are the *simulators* and the *implementations* of membrane systems. A "standard" application in biology consists of writing a (mathematical) model, then simulating its evolution on a computer; this means writing a program, a software support, then making experiments (tuning parameters, checking the evolution in time, modifying certain features of the model, checking the evolution in time, and repeating these steps in a dialogue between biologist and computer scientist and between user and software producer), until adequately covering the modeled reality and until obtaining relevant insights about it.

The importance of having at hand an efficient, reliable, and easy to use computer support—both software and hardware—is obvious in this framework. This means the importance of *simulation* and of *implementation* of membrane systems.

The difference between the two is apparent—and we have to mention from the very beginning that at this moment there is no real bio-implementation of a P system, no lab "wet" simulation (unlike, for instance, the case of *DNA computing*, which started in 1987 by a theoretical model, the so-called splicing operation, introduced by Tom Head, but got an essential impulse in 1994, by the Leonard Adleman lab experiment of computing in a test tube, using DNA molecules). There are, however, many programs for simulating various types of P systems, using a large variety of programming languages and programming techniques—on the top of all these,

there also exists a dedicated programming language, *P-lingua*, elaborated by the membrane computing research group at Seville University, Spain.

In several research groups, a series of attempts were made to implement P systems on a dedicated hardware (the big challenges are related to the implementation of the distribution and, mainly, of the parallelism of the model), one specially designed for this purpose or adapted to that (such as existing parallel hardware, like GPUs—graphic processing units).

The present book provides details about all these directions of investigation—and still more, as it starts with general considerations about the difficulties encountered when trying to simulate or implement (the borderline/difference between the two are not always sharp) a P system. It also describes the possibility of automatic designing of a P system, proposes a workbench for verifying the obtained models, and discusses a series of applications.

This is a really necessary and timely book, written by a team of professionals, among the most active and highly knowledgeable experts in membrane computing, from all three points of view mentioned in the title—theory, applications, implementations. The reader interested in modeling real-life processes, especially of a discrete, compartmental/distributed type, or in developing tools for that, software or hardware alike, will surely benefit from exploring this book.

Bucharest, Romania Gheorghe Păun
November 25, 2020

Preface

Membrane computing is a branch of natural computing, investigating computational models, called *membrane systems* or *P systems*, inspired by the structure and functioning of the biological cell. This computing paradigm was initiated by Gheorghe Păun in 1998. The membrane computing community has succeeded to achieve during its more than 20 years' history a set of landmark successes: the establishment of the *International Membrane Computing Society* (IMCS); the organisation of four regular conference/workshop events, namely *European Branch of International Conference on Membrane Computing* (ECMC), *Asian Branch of International Conference on Membrane Computing* (ACMC), *Brainstorming Week on Membrane Computing* (BWMC), and *Chinese Workshop on Membrane Computing* (CWMC); and the gestation and birth of two periodic publications, *Journal of Membrane Computing* (JMC)—four issues per year—and IMCS Bulletin—two issues per year. IMCS awards three yearly IMCS Prizes: The PhD Thesis of the Year, The Theoretical Result of the Year and The Application of the Year.

This book presents for the first time to the international community a set of robust, efficient, reliable and easy-to-use tools supporting membrane computing models. As numerous theoretical results and a wide range of applications have been published in several comprehensive monographs or collective books

G. Ciobanu, M.J. Pérez-Jiménez, Gh. Păun (eds.), *Applications of Membrane Computing*, in Natural Computing Series, Springer, 2006.
P. Frisco, M. Gheorghe, M.J. Pérez-Jiménez (eds.), *Applications of Membrane Computing in Systems and Synthetic Biology*, in Emergence, Complexity and Computation Series, Springer, 2014.
Gh. Păun, *Membrane Computing—An Introduction*, Springer, 2002.
Gh. Păun, G. Rozenberg, A. Salomaa (eds.), *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2010.
G. Zhang, M.J. Pérez-Jiménez, M. Gheorghe. *Real-life applications with membrane computing*, Springer, 2017.
G. Zhang, J. Cheng, T. Wang, X. Wang, J. Zhu. *Membrane Computing: Theory and Applications*, Science China Press, 2015.

this book aims to present the most recent and significant implementation models, algorithms and platforms of membrane computing models on a variety of software

and hardware platforms, and to describe the most relevant applications, facilitating a better and deep understanding on how the tools are used in building, experimenting with and analysing membrane computing models of complex problems arising in robotics, automatic design of P systems, ecosystem modelling, systems and synthetic biology, and bioinformatics.

The chapters covered in this monograph provide a clear image of the depth and breadth of the software and hardware implementations of membrane systems.

- In Chap. 1, *Introduction*: An overview of membrane computing is provided. Software and hardware implementation development of P systems are presented. The challenging problems of P systems implementation are discussed.
- In Chap. 2, *P Systems Implementation on P-Lingua Framework*: P-Lingua framework, the most widely used product for the specification and simulation of different types of P systems, is described. A high-level tool for virtual experiments, membrane computing simulator (MeCoSim) is also presented.
- In Chap. 3, *Applications of Software Implementations of P Systems*: Automatic design of cell-like P systems and spiking neural P systems are discussed to address the programmability issue of membrane computing models. The methodologies for modelling real ecosystems and mobile robots motion planning are also presented.
- In Chap. 4, *Infobiotics Workbench: An In Silico Software Suite for Computational Systems Biology*: The Infobiotics Workbench, an integrated software suite developed for computational systems biology, is presented. The tool is built upon stochastic P systems, a probabilistic extension of P systems, as modelling framework. The platform utilises computer-aided modelling and analysis of biological systems through simulation, verification and optimisation.
- In Chap. 5, *Molecular Physics and Chemistry in Membranes: the Java Environment for Nature-Inspired Approaches (JENA)*: The JENA is described as a modular, configurable and extendable platform towards a *virtual laboratory* and a *virtual cell* complementing more abstract and more idealised approaches in membrane computing. An introduction to JENA with its features and capabilities from the user's perspective and from a technical point of view is given. Four illustrative case studies are used to demonstrate JENA's practicability and descriptive capacity.
- In Chap. 6, *P Systems Implementation on Graphics Processing Units (GPUs)*: The concepts behind GPU computing and a taxonomy of GPU-based simulators, generic, specific and adaptive simulation, are introduced.
- In Chap. 7, *P Systems Implementation on Field-Programmable Gate Arrays (FPGA)*: Different existing implementations of P systems using FPGA hardware are presented. The strong and the weak points of each implementation is given. A particular attention is given to the latest implementation of generalised numerical P systems that considers many advanced techniques. A discussion about the challenges and the necessity of an FPGA implementation is finally performed.

- In Chap. 8, *Applications of Hardware Implementation of P Systems*: Enzymatic numerical P system (ENPS) based robot controllers and path planning algorithm are implemented in FPGA, achieving a speedup of 10^5 and 10^4 order of magnitude compared to software simulation. FPGA-hardened (E)NPS in this research can be regarded as a heterogeneous multicore processor since membranes inside work as processing units which possess different functions.

This book provides comprehensive descriptions of software and hardware tools, making it a valuable resource for anyone interested in membrane computing models, and it will be of particular interest to researchers looking for implementation methodology of membrane systems and also a variety of computing models in natural computing. The readers can get benefits from this book in the aspects of software or hardware implementation ideas of parallel distributed computing models, software/hardware development ideas and skills, the link procedure between computing models with applications.

Chengdu, China — Gexiang Zhang
Sevilla, Spain — Mario J. Pérez-Jiménez
Sevilla, Spain — Agustín Riscos-Núñez
Paris, France — Sergey Verlan
Bradford, UK — Savas Konur
Jena, Germany — Thomas Hinze
Bradford, UK — Marian Gheorghe

Acknowledgments

The first thank is given to the father of membrane computing, Gheorghe Păun, for his persistently strong support and perfect foreword, especially on the occasion of his 70th birthday.

The conception of this book originates from the friendly and fruitful collaborations of our five teams (Chengdu, Sevilla, Paris, Bradford and Jena) in the past more than 10 years. Many contributors could not be included in the author list and are therefore acknowledged below.

The authors gratefully acknowledge the excellent work and outstanding contributions of Ignacio Pérez-Hurtado, David Orellana-Martín, Miguel Ángel Martínez-del-Amor, and Luis Valencia-Cabrera to the Sevilla team; Prithwineel Paul, Zeyi Shang, Jianping Dong, Zhu Ou, and Xiaoli Huang to the Chengdu team; Jonathan Blakes, Jamie Twycross, Natalio Krasnogor, Francisco Jose Romero-Campero, and Laurenţiu Mierlă to Infobiotics Workbench; 35 students up till now to the JENA software system implementation presented in Chap. 5; Anthony Aguillon, Benjamin Förster, Gerd Grünert, Hendrik Happe, Florian Höch, Sinan Kaya, Korcan Kirkici, Alexander Melcher, Daniel Noelpp, Jonas Pilot, and Lea Weber who claim the greatest share to turn the project into major success.

The work of Gexiang Zhang and Sergey Verlan is supported by the National Natural Science Foundation of China (61972324, 61672437, 61702428, 61373047, 61170016), Sichuan Science and Technology Program (2018GZ0185, 2018GZ0086), Beijing Advanced Innovation Center for Intelligent Robots and Systems (2019IRS14), New Generation Artificial Intelligence Science and Technology Major Project of Sichuan Province (2018GZDZX0043), and Artificial Intelligence Key Laboratory of Sichuan Province (2019RYJ06). The work of Savas Konur is supported by EPSRC research grant EP/R043787/1. The work of Mario Pérez-Jiménez, Agustín Riscos-Núñez and the rest of the Sevilla team is supported by research project TIN2017-89842-P (MABICAP), co-financed by Ministerio de Ciencia e Innovación of Spain, through the *Agencia Estatal de Investigación* (AEI), and by *Fondo Europeo de Desarrollo Regional* (FEDER) of the European Union.

Contents

1 Introduction 1
1.1 Membrane Computing Overview 1
1.2 Software Implementation of P Systems 2
1.3 Hardware Implementation of P Systems 3
1.4 Challenges of P Systems Implementation 4
1.5 Concluding Remarks 5
References 6

2 P Systems Implementation on P-Lingua Framework 11
2.1 Introduction 11
2.2 P-Lingua Language 12
2.2.1 P System Models 12
2.2.2 Membrane Structure 14
2.2.3 Initial Multisets 15
2.2.4 P System Rules 15
2.3 Simulation Algorithms 17
2.4 Membrane Computing Simulator (MeCoSim) 19
2.4.1 Primary goals 20
2.4.2 Main Functional Components 22
2.5 Conclusion 27
References 27

3 Applications of Software Implementations of P Systems 31
3.1 Introduction 31
3.2 Automatic Design of Cell-Like P Systems with P-Lingua 32
3.2.1 Preliminaries 32
3.2.2 Automatic Design of P Systems with an Elitist Genetic Algorithm 34
3.2.3 Automatic Design of P Systems with a Permutation Penalty Genetic Algorithm 40
3.3 Automatic Design of Spiking Neural P Systems with P-Lingua 49
3.4 Modelling Real Ecosystems with MeCoSim 55
3.4.1 Problem Description 55

3.5 Robot Motion Planning 59
3.5.1 Problem Definition 62
3.5.2 Path Planning for Mobile Robots 63
3.5.3 Rapidly-Exploring Random Tree (RRT) Algorithm 63
3.6 Conclusion 64
References 64

4 Infobiotics Workbench: An In Silico Software Suite for Computational Systems Biology 71
4.1 Introduction 71
4.2 Stochastic P Systems 72
4.3 Software Description 77
4.3.1 Simulation 77
4.3.2 Verification 80
4.3.3 Optimization 82
4.4 Case Studies 85
4.4.1 Pulse generator 85
4.4.2 Repressilator 90
4.5 KPWorkbench: A Qualitative Analysis Tool 93
4.6 Next-Generation Infobiotics for Synthetic Biology 94
4.7 Conclusion 95
References 95

5 Molecular Physics and Chemistry in Membranes: The Java Environment for Nature-Inspired Approaches (JENA) 101
5.1 Introduction 101
5.2 JENA at a Glance and Its Descriptive Capacity 105
5.2.1 Atoms, Ions, Molecules, and Particles 108
5.2.2 Vessels and Delimiters 117
5.2.3 Brownian Motion and Thermodynamics 121
5.2.4 Chemical Reactions by Effective Collisions and by Spontaneous Decay 125
5.2.5 Applying External Forces 130
5.2.6 Active Membranes and Dynamical Delimiters 134
5.2.7 Simulation, Monitoring, Logging, and Analyses 136
5.3 JENA Source Code Design 139
5.4 Selection of JENA Case Studies 141
5.4.1 Chemical Lotka-Volterra Oscillator 142
5.4.2 Electrophoresis 144
5.4.3 Centrifugation 149
5.4.4 Neural Signal Transduction Across Synaptic Cleft 152
5.5 Conclusions and Prospectives 158
References 160

6 P Systems Implementation on GPUs 163
6.1 Introduction 163
6.2 GPU Computing 164
6.2.1 The Graphics Processing Unit 164
6.2.2 CUDA Programming Model 166
6.2.3 GPU Architecture 168
6.2.4 Good Practices 169
6.3 Generic Simulations 170
6.3.1 Definition 170
6.3.2 Simulating P Systems with Active Membranes 171
6.3.3 Simulating Population Dynamics P Systems 182
6.4 Specific Simulations 189
6.4.1 Definition 190
6.4.2 Simulating a SAT Solution with Active Membrane P Systems 191
6.4.3 Simulating a SAT Solution with Tissue P Systems 198
6.5 Adaptive Simulations 209
6.5.1 Definition 209
6.5.2 Simulating Population Dynamics P Systems 209
6.6 Conclusions 212
References 213

7 P Systems Implementation on FPGA 217
7.1 Introduction 217
7.2 FPGA Hardware 218
7.3 Generalized Numerical P Systems (GNPS) 221
7.3.1 Formal Definition 221
7.3.2 Basic Variant 225
7.3.3 Historical Remarks 228
7.4 Implementing GNPS on FPGA 230
7.5 FPGA Implementations of Other Models of P Systems 234
7.5.1 Petreska and Teuscher Implementation 234
7.5.2 Nguyen Implementation 235
7.5.3 Quiros and Verlan Implementation 237
7.5.4 Comments 238
7.6 Discussion 239
7.7 Conclusion 240
References 241

8 Applications of Hardware Implementation of P Systems 245
8.1 Introduction 245
8.2 Robot Membrane Controllers with FPGA Implementation 246
8.2.1 Numerical P Systems-Based Membrane Controllers on FPGA 246

8.2.2 Enzymatic Numerical P Systems (ENPS)-Based Membrane Controllers on FPGA 253
8.2.3 GNPS-Based Membrane Controllers on FPGA 254
8.3 Robot Path Planning with FPGA Implementation 256
8.3.1 RRT Algorithm 258
8.3.2 Arithmetic Units Design 261
8.3.3 Enzymatic Numerical P System Rapid-Exploring Random Tree Register Transfer Level (ENPS-RRT RTL) Model Design 265
8.3.4 ENPS-RRT on FPGA 268
8.4 Conclusion 272
References 273

Correction to: Membrane Computing Models: Implementations C1

Index 277

Acronyms

3-COL	3-CoLor (problem)
AC	Alternating Current
AG	Average Generation
BBB	Binomial Block Based (algorithm)
CLBs	Configurable Logic Blocks
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DC	Direct Current
DCBA	Direct Distribution Based on Consistent Blocks Algorithm
DNA	Deoxyribonucleic Acid
DND	Direct Non-Deterministic Distribution
DNDP	Direct Non-Deterministic Algorithm with Probabilities
DSL	Domain Specific Language
DSP	Digital Signal Processing
ENPS	Enzymatic Numerical P Systems
FFs	Flip-Flops
FPGA	Field Programmable Gate Array
GAPE	Genetic Algorithm with the Permutation Encoding Technique
GNPS	Generalized Numerical P Systems
GPU	Graphics Processing Unit
HAM-CYCLE	Hamiltonian CYCLE
HDL	Hardware Description Language
HPC	High Performance Computing
IBL	Infobiotics Language
IBW	Infobiotics Workbench
ILA	Integrated Logic Analyzer
JENA	Java Environment for Nature-Inspired Approaches
JGAP	Java Genetic Algorithm Package
KP	Kernel P (Systems)
LC	Logic Cell
LFSR	Linear Feedback Shift Register
LHS	Left-Hand Side
LPP	Lattice Population P (Systems)
LUTs	Look-Up Tables

MC	Membrane Computing
MCSS	Multi-compartmental Stochastic Simulation
MeCoSim	Membrane Computing Simulator
MX	Multiplexers
NoFE	Number of Function Evaluations
NP problem	Nondeterministic Polynomial Time Problem
NPS	Numerical P systems
PDP	Population Dynamics P (System)
PID	Proportional Integral Derivative Controller
PPGA	Permutation Penalty Genetic Algorithm
PRNG	Pseudo Random Number Generator
PSNP	SNP Systems with Polarization
PSPACE	Polynomial Space (complete)
QIEA	Quantum-Inspired Evolutionary Algorithm
RHS	Right-Hand Side
RNA	RiboNucleic Acid
RRTs	Rapidly Exploring Random Trees
RTL	Register-Transfer Level
SAT	Satisfiability (problem)
SBML	Systems Biology Markup Language
SIMD	Single Instruction Multiple Data
SN	Spiking Neural (P system)
SNP	Spiking Neural P (System)
SP	Stochastic P (Systems)
SR	Success Rate
SRSim	Spatial Rules Simulator
SSA	Stochastic Simulation Algorithm
SNUPS	Simulator of Numerical P Systems
TRNG	True Random Number Generator
UART	Universal Asynchronous Receiver Transmitter
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit