

N. Nedjah, E. Alba, L. de Macedo Mourelle (Eds.)

---

Parallel Evolutionary Computations

## Studies in Computational Intelligence, Volume 22

### Editor-in-chief

Prof. Janusz Kacprzyk  
Systems Research Institute  
Polish Academy of Sciences  
ul. Newelska 6  
01-447 Warsaw  
Poland  
E-mail: kacprzyk@ibspan.waw.pl

---

Further volumes of this series  
can be found on our homepage:  
[springer.com](http://springer.com)

Vol. 7. Bruno Apolloni, Ashish Ghosh, Ferda  
Alpaslan, Lakhmi C. Jain, Srikanta Patnaik  
(Eds.)  
*Machine Learning and Robot Perception*,  
2005  
ISBN 3-540-26549-X

Vol. 8. Srikanta Patnaik, Lakhmi C. Jain,  
Spyros G. Tzafestas, Germano Resconi,  
Amit Konar (Eds.)  
*Innovations in Robot Mobility and Control*,  
2006  
ISBN 3-540-26892-8

Vol. 9. Tsau Young Lin, Setsuo Ohsuga,  
Churn-Jung Liao, Xiaohua Hu (Eds.)  
*Foundations and Novel Approaches in Data  
Mining*, 2005  
ISBN 3-540-28315-3

Vol. 10. Andrzej P. Wierzbicki, Yoshiteru  
Nakamori  
*Creative Space*, 2005  
ISBN 3-540-28458-3

Vol. 11. Antoni Ligeza  
*Logical Foundations for Rule-Based  
Systems*, 2006  
ISBN 3-540-29117-2

Vol. 12. Jonathan Lawry  
*Modelling and Reasoning with Vague  
Concepts*, 2006  
ISBN 0-387-29056-7

Vol. 13. Nadia Nedjah, Ajith Abraham,  
Luiza de Macedo Mourelle (Eds.)  
*Genetic Systems Programming*, 2006  
ISBN 3-540-29849-5

Vol. 14. Spiros Sirmakessis (Ed.)  
*Adaptive and Personalized Semantic Web*,  
2006  
ISBN 3-540-30605-6

Vol. 15. Lei Zhi Chen, Sing Kiong Nguang,  
Xiao Dong Chen  
*Modelling and Optimization of  
Biotechnological Processes*, 2006  
ISBN 3-540-30634-X

Vol. 16. Yaochu Jin (Ed.)  
*Multi-Objective Machine Learning*, 2006  
ISBN 3-540-30676-5

Vol. 17. Te-Ming Huang, Vojislav Kecman,  
Ivica Kopriva  
*Kernel Based Algorithms for Mining Huge  
Data Sets*, 2006  
ISBN 3-540-31681-7

Vol. 18. Chang Wook Ahn  
*Advances in Evolutionary Algorithms*, 2006  
ISBN 3-540-31758-9

Vol. 19. N. Ichalkaranje, A. Ichalkaranje,  
L.C. Jain (Eds.)  
*Intelligent Paradigms for Assistive and  
Preventive Healthcare*, 2006  
ISBN 3-540-31762-7

Vol. 20. Wojciech Penczek, Agata Pórola  
*Advances in Verification of Time Petri Nets  
and Timed Automata*, 2006  
ISBN 3-540-32869-6

Vol. 21. Cândida Ferreira  
*Gene Expression Programming:  
Mathematical Modeling by an Artificial  
Intelligence*, 2006  
ISBN 3-540-32796-7

Vol. 22. N. Nedjah, E. Alba, L. de Macedo  
Mourelle (Eds.)  
*Parallel Evolutionary Computations*, 2006  
ISBN 3-540-32837-8

N. Nedjah  
E. Alba  
L. de Macedo Mourelle  
(Eds.)

# Parallel Evolutionary Computations

 Springer

Dr. Nadia Nedjah  
Department of System Engineering  
and Computation  
Faculty of Engineering  
State University of Rio de Janeiro  
Rua São Francisco Xavier, 524, 5o. andar  
Maracanã, CEP 20559-900  
Rio de Janeiro, RJ  
Brazil  
E-mail: nadia@eng.uerj.br

Professor Enrique Alba  
Universidad de Málaga  
Depto. of Lenguajes y Ciencias de la  
Computación  
Campus de Teatinos  
29071 Málaga, Spain  
E-mail: eat@lcc.uma.es

Dr. Luiza de Macedo Mourelle  
Department of System Engineering  
and Computation  
Faculty of Engineering  
State University of Rio de Janeiro  
Rua São Francisco Xavier, 524, 5o. andar  
Maracanã, CEP 20559-900  
Rio de Janeiro, RJ  
Brazil  
E-mail: ldmm@eng.uerj.br

Library of Congress Control Number: 2006921792

ISSN print edition: 1860-949X  
ISSN electronic edition: 1860-9503  
ISBN-10 3-540-32837-8 Springer Berlin Heidelberg New York  
ISBN-13 978-3-540-32837-7 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, reuse of illustrations, recitation, broadcasting, reproduction on microfilm 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 for prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media  
springer.com  
© Springer-Verlag Berlin Heidelberg 2006  
Printed in The Netherlands

The use of general descriptive names, registered names, trademarks, 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.

Typesetting: by the authors and TechBooks using a Springer L<sup>A</sup>T<sub>E</sub>X macro package  
Printed on acid-free paper SPIN: 11544210 89/TechBooks 5 4 3 2 1 0

*To the memory of my father Ali and my beloved mother Fatiha,*

Nadia Nedjah

*Thanks to my wife Ana,*

Enrique Alba

*To the memory of my beloved father Luiz and my mother Neuza,*

Luiza Mourelle

---

## Preface

Evolutionary algorithms and their related computations are solver systems which use computational models inspired in Darwinian natural selection processes as a key element in their design and implementation. In general, evolutionary computation (EC) is used to solve *NP*-hard problems which cannot be solved with other tools because of their intrinsic difficulty, high dimensionality or incomplete definition. In practice, EC is composed of a set of different families of algorithms that iteratively improve a set of tentative solutions to obtain an optimal or quasi-optimal solution to the problem. Therefore, evolutionary algorithms require sometimes a massive computational effort to yield efficient and competitive solutions to real-size engineering problems which would otherwise rest unsolved today.

This book focuses on the aspects related to the parallelization of evolutionary computations, such as parallel genetic operators, parallel fitness evaluation, distributed genetic algorithms, and parallel hardware implementations, as well as on the impact of parallel EC on several applications.

The book is divided into four parts. The first part deals with a clear software-like and algorithmic vision on parallel evolutionary optimizations. The second part is about hardware implementations of genetic algorithms, a valuable topic which is hard to find in the present literature. The third part treats the problem of distributed evolutionary computation and presents three interesting applications wherein parallel EC new ideas are featured. Finally, the last part deals with the up-to-date field of parallel particle swarm optimization to illustrate the intrinsic similarities and potential extensions to techniques in this domain.

The goal of this volume has been to offer a wide spectrum of sample works developed in leading research throughout the world about parallel implementations of efficient techniques at the heart of computational intelligence. The book should be useful both for beginners and experienced researchers in the field of computational intelligence.

## Part I: Parallel Evolutionary Optimization

In Chapter 1, which is entitled *A Model for Parallel Operators in Genetic Algorithms*, the authors analyze a model for applying crossover and varying mutation separately in parallel. The authors focus on the gains of performance that can be achieved from the concurrent application of variation operators with different and complementary roles. An important aim of this chapter is to show that, for the actual parallelization of operators to be meaningful, appropriate models for the concurrent application of operators should be devised first.

In Chapter 2, which is entitled *Parallel Evolutionary Multiobjective Optimization*, the authors' first goal is to provide the reader with a wide overview of the literature on parallel evolutionary algorithms for multiobjective optimization. Also, the authors include an experimental study wherein *pPAES*, a parallel evolutionary algorithm for multiobjective optimization based on the Pareto Archived Evolution Strategy (PAES) is developed and analyzed. The obtained results show that *pPAES* is a promising option for solving multiobjective optimization problems.

## Part II: Parallel Hardware for Genetic Algorithms

In Chapter 3, which is entitled *A Reconfigurable Parallel Hardware for Genetic Algorithms*, the authors propose a massively parallel architecture for hardware implementation of genetic algorithms. The authors claim that this design is quite innovative as it provides a viable solution to the fitness computation problem, which depends heavily on the problem-specific knowledge. The proposed architecture is completely independent of such specifics. It implements the fitness computation using a neural network. The hardware implementation of the neural network is stochastic and thus minimises the required hardware area without much increase in response time. The authors also demonstrate the characteristics of the proposed hardware and compare it to existing ones.

In Chapter 4, which is entitled *Reconfigurable Computing and Parallelism for Implementing and Accelerating Evolutionary Algorithms*, the authors present two instances of hardware implementation of evolutionary algorithms. Both instances provide distinct points of view on how to apply reconfigurable computing technology to increase algorithm efficiency. The cases considered are genetic algorithms and one parallel evolutionary algorithm. In particular, the authors illustrate through an experimental study the performance of the presented hardware implementations using the Travelling Salesman Problem.

### Part III: Distributed Evolutionary Computation

In Chapter 5, which is entitled *Performance of Distributed GAs on DNA Fragment Assembly*, the authors present results on analyzing the behaviour of a parallel distributed genetic algorithm over different LAN technologies. Their main goal is to offer a study on the potential impact in the search mechanics when shifting between LANs: a Fast Ethernet network, a Gigabit Ethernet network, and a Myrinet network. They also study the importance of several parameters of the migration policy. The author use the DNA fragment assembly problem to show the actual power and utility of the proposed distributed technique.

In Chapter 6, which is entitled *On Parallel Evolutionary Algorithms on the Computational Grid*, the authors analyze the major traditional parallel models of evolutionary algorithms. The objective consists of adapting the parallel models to grids taking into account the characteristics of such execution environments in terms of volatility, heterogeneity, large scale and multi-administrative domain. The authors give an overview of such frameworks and present a case study related to ParadisEO-CMW which is a porting of ParadisEO onto Condor and MW allowing a transparent deployment of the parallel EAs on grids.

In Chapter 7, which is entitled *Parallel Evolutionary Algorithms on Consumer-Level Graphics Processing Unit*, the author propose to implement a parallel evolutionary algorithm on consumer-level Graphics Processing Unit. They perform experiments to compare the proposed parallel evolutionary algorithm with an ordinary one and demonstrate that the former is much more effective than the latter. The authors claim that as consumer-level graphics processing units are already widely available and installed on personal computers and they are easy to use and manage, more people will be able to use our parallel algorithm to solve their problems encountered in real-world applications.

### Part IV: Parallel Particle Swarm Optimization

In Chapter 8, which is entitled *Intelligent Parallel Particle Swarm Optimization Algorithms*, the authors present a parallel version of the particle swarm optimization (PPSO) algorithm together with three communication strategies which can be used according to the independence of the data. They discuss some communication strategies for PPSO, which can be used according to the strength of the correlation of parameters. Experimental results confirm the superiority of the PPSO algorithms.

In Chapter 9, which is entitled *Parallel Ant Colony Optimization for 3D Protein Structure Prediction using the HP Lattice Model*, the authors introduce a novel method of solving the Hydrophobic-Hydrophilic (HP) protein



folding problem in both two and three dimensions using Ant Colony Optimizations and a distributed programming paradigm. They claim that tests across a small number of processors indicate that the multiple colony distributed ACO (MACO) approach outperforms single colony implementations and that experimental results also demonstrate that the proposed algorithms perform well in terms of network scalability.

We are very much grateful to the authors of this volume and to the reviewers for their tremendous service by critically reviewing the chapters. The editors would also like to thank Prof. Janusz Kacprzyk, the editor-in-chief of the Studies in Computational Intelligence Book Series and Dr. Thomas Ditzinger from Springer-Verlag, Germany for their editorial assistance and excellent collaboration to produce this scientific work. We hope that the reader will share our excitement on this **Parallel Evolutionary Computations** volume and will find it useful.

Brazil, Spain  
January 2006

*Nadia Nedjah*  
*Enrique Alba*  
*Luiza M. Mourelle*

---

# Contents

---

## Part I Parallel Evolutionary Optimization

---

### 1 A Model for Parallel Operators in Genetic Algorithms

<i>Hernán Aguirre and Kiyoshi Tanaka</i> .....	3
1.1 Introduction .....	3
1.2 Implicit Parallel Operators in Canonical and Conventional Varying Mutation GAs .....	6
1.3 A Model of Parallel Varying Mutation GA (GA-SRM) .....	7
1.3.1 Explicit Parallel Operators .....	8
1.3.2 Extinctive Selection .....	8
1.3.3 Mutation Rate Control .....	9
1.3.4 Mutation Strategy .....	11
1.4 0/1 Multiple Knapsacks Problems .....	12
1.5 Studying the Structure of the Parallel Varying Mutation GA-SRM.....	14
1.5.1 Simple GA and Extinctive Selection .....	14
1.5.2 Contribution of Parallel Genetic Operators and Extinctive Selection .....	16
1.5.3 Results on Classes of 0/1 Multiple Knapsack Problems .....	17
1.6 Comparing Conventional and Parallel Varying Mutation Models .....	18
1.6.1 Deterministic Varying Mutation .....	19
1.6.2 Self-Adaptive Varying Mutation .....	21
1.7 Distributed GA with Parallel Varying Mutation .....	23
1.7.1 Problem Difficulty .....	25
1.8 Summary .....	29
References .....	29

**2 Parallel Evolutionary Multiobjective Optimization**

<i>Francisco Luna, Antonio J. Nebro, Enrique Alba</i> .....	33
2.1 Introduction .....	33
2.2 Multiobjective Optimization .....	35
2.2.1 Theoretical Background .....	35
2.2.2 A Survey of Parallel MOEAs .....	36
2.2.3 Other Parallel Algorithms for Solving MOPs .....	42
2.3 A Parallel MOEA: pPAES .....	43
2.3.1 (1 + 1)-PAES .....	43
2.3.2 pPAES .....	44
2.3.3 Experiments .....	45
2.4 Summary .....	49
References .....	50

---

**Part II Parallel Hardware for Genetic Algorithms**


---

**3 A Reconfigurable Parallel Hardware for Genetic Algorithms**

<i>Nadia Nedjah, Luiza de Macedo Mourelle</i> .....	59
3.1 Introduction .....	59
3.2 Principles of Genetic Algorithms .....	60
3.3 Overall Architecture	
for the Hardware Genetic Algorithm .....	61
3.4 Detailed Component Architectures .....	61
3.4.1 Shared Memory for Generational Population .....	62
3.4.2 Random Number Generator .....	62
3.4.3 Selection Component .....	63
3.4.4 Genetic Operator's Components .....	65
3.4.5 Fitness Evaluation Component .....	66
3.5 Performance Results .....	67
3.6 Summary .....	68
References .....	69

**4 Reconfigurable Computing and Parallelism for Implementing and Accelerating Evolutionary Algorithms**

<i>Miguel A. Vega Rodríguez, Juan A. Gómez Pulido, Juan M. Sánchez Pérez, José M. Granado Criado, Manuel Rubio del Solar</i> .....	71
4.1 Introduction .....	71
4.2 Reconfigurable Computing and FPGA .....	72
4.3 Implementing a Genetic Algorithm	
for Solving the TSP Using Parallelism and FPGAs .....	74
4.3.1 The TSP .....	75
4.3.2 The Genetic Algorithm Used .....	76
4.3.3 Hardware Implementation Using Parallelism and FPGAs .....	77

4.3.4 Results .....	79
4.3.5 Conclusions .....	83
4.4 Hardware Acceleration	
of a Parallel Evolutionary Algorithm .....	84
4.4.1 The Problem: Modeling and Predicting Time Series .....	85
4.4.2 A Parallel Evolutionary Heuristic .....	86
4.4.3 Design of a Reconfigurable Processor .....	87
4.4.4 Experimental Results .....	89
4.4.5 Conclusion .....	91
4.5 Summary .....	91
References .....	92

---

**Part III Distributed Evolutionary Computation**

---

**5 Performance of Distributed GAs on DNA Fragment**

**Assembly**

<i>Enrique Alba, Gabriel Luque</i> .....	97
5.1 Introduction .....	97
5.2 DNA Fragment Assembly Problem .....	98
5.2.1 DNA Sequencing Process .....	99
5.3 Distributed Genetic Algorithms .....	101
5.4 DNA Fragment Assembly Using a Distributed GA .....	103
5.4.1 The MALLBA Project .....	105
5.5 Experimental Results .....	105
5.6 Summary .....	113
References .....	114

**6 On Parallel Evolutionary Algorithms  
on the Computational Grid**

<i>N. Melab, E-G. Talbi and S. Cahon</i> .....	117
6.1 Introduction .....	117
6.2 Principles of Evolutionary Algorithms .....	118
6.3 Parallel EAs on the Computational Grid .....	119
6.3.1 The Island Model .....	121
6.3.2 The Parallel Evaluation of the Population .....	124
6.3.3 The Parallel Evaluation of a Single Solution .....	125
6.4 Frameworks for Grid-based EAs – The ParadisEO-CMW Case ...	127
6.4.1 ParadisEO: an Extended EO .....	127
6.4.2 ParadisEO-CMW: Grid-enabled ParadisEO .....	128
6.5 Conclusion .....	130
References .....	131

<b>7 Parallel Evolutionary Algorithms on Consumer-Level Graphics Processing Unit</b>	
<i>Tien-Tsin Wong, Man Leung Wong</i> .....	133
7.1 Introduction .....	133
7.2 Parallel and Distributed Evolutionary Algorithms .....	134
7.3 Graphics Processing Unit .....	136
7.4 Data Organization .....	139
7.5 Evolutionary Programming on GPU .....	140
7.5.1 Mutation and Reproduction .....	141
7.5.2 Fitness Value Evaluation .....	143
7.5.3 Competition and Selection .....	143
7.6 Experimental Results and Visualization .....	146
7.6.1 Experimental Results .....	146
7.6.2 Visualization .....	151
7.7 Summary .....	154
References .....	154

---

## Part IV Parallel Particle Swarm Optimization

---

<b>8 Intelligent Parallel Particle Swarm Optimization Algorithms</b>	
<i>Shu-Chuan Chu, Jeng-Shyang Pan</i> .....	159
8.1 Introduction .....	159
8.2 Parallel Particle Swarm Optimization .....	165
8.3 Experimental Results .....	170
8.4 Conclusions .....	174
References .....	174
<b>9 Parallel Ant Colony Optimization for 3D Protein Structure Prediction using the HP Lattice Model</b>	
<i>Daniel Chu, Albert Zomaya</i> .....	177
9.1 Introduction .....	177
9.2 Background .....	178
9.2.1 Protein Structures .....	178
9.2.2 De Novo Modelling .....	179
9.2.3 Comparative Modelling .....	179
9.2.4 Molecular Dynamics Simulations .....	179
9.2.5 Hydrophobic-Hydrophilic Lattice Model .....	180
9.2.6 Existing Algorithms for the HP Lattice Model .....	180
9.2.7 Evolutionary Algorithms .....	181
9.2.8 Hydrophobic Zipper (HZ) .....	182
9.2.9 Contact Interaction .....	182
9.3 Ant Colony Optimization .....	183
9.3.1 Pheremone Matrix .....	184
9.3.2 Local Search .....	184

9.3.3	Population Based ACO	185
9.3.4	Multi Colony ACO (MACO)	185
9.4	ACO Implementation	185
9.4.1	Construction Phase	186
9.4.2	Heuristics	186
9.4.3	Coding	187
9.4.4	Local Search	187
9.4.5	Pheremone Updates	187
9.4.6	Single Process Single Colony	188
9.4.7	Distributed Single Colony	188
9.4.8	Distributed Multi Colony with Circular Exchange of Migrants	189
9.4.9	Distributed Mult Colony with Pheremone Matrix Sharing	189
9.5	Results	189
9.5.1	Testing Methodology	189
9.5.2	Emperical Results	190
9.5.3	Convergence Speed Differences	190
9.5.4	Scalability	191
9.5.5	Network Traffic vs Protein Length	194
9.6	Conclusion and Future Work	195
9.6.1	Solution Quality	195
9.6.2	Long Protein Chains	196
9.6.3	Variable Colony Sizes	196
9.6.4	Investigation of Fitness Landscape	196
	References	197
	<b>Subject Index</b>	199
	<b>Author Index</b>	201

---

## List of Figures

1.1	Canonical Genetic Algorithm .....	6
1.2	Genetic Algorithm with Conventional Varying Mutation.....	6
1.3	GA with Parallel Varying Mutation.....	10
1.4	Deterministic Hyperbolic Schedule for Mutation Rate Control..	10
1.5	Knapsack Problem .....	13
1.6	Effect of Extinctive Selection on a Simple GA ( $m = 30$ , $n = 100$ , $\phi = 0.25$ ).....	15
1.7	Transition from GA(50,100) to GA-SRM(50,100) at various fractions of $T$ .....	17
1.8	Transitions from GA- SRM(50,100) to either all CM(50,100) or all SRM(50,100) regimes at $\ell = \{2, 3\}$ .....	17
1.9	Reducing the feasible region $\phi = \{0.75, 0.50, 0.25\}$ , $m = 30$ , $n = 100$ .....	18
1.10	Increasing the number of constraints $m = \{5, 10, 30\}$ , $n = 100$ , $\phi = 0.25$ .....	18
1.11	Increasing the search space $2^n$ , $n = \{100, 250, 500\}$ , $m = 30$ , $\phi = 0.25$ .....	18
1.12	Deterministic Varying Mutation ( $m = 30$ , $n = 100$ , $\phi = 0.25$ ) ..	19
1.13	Self-Adaptive Varying Mutation $p_m^{(t=0)}(i) = p_m^{max}$ ( $m = 30$ , $n = 100$ , $\phi = 0.25$ ).....	22
1.14	Convergence Velocity and Average Number of Flipped Bits ( $m = 30$ , $n = 100$ , $\phi = 0.25$ ). $p_m^{(t=0)} = 0.5$ for hGA and GA-hM. $p_m^{(t=0)}(i) = r$ and $[1/n, 0.5]$ for sGA and GA-SM .....	23
1.15	+1+2 communication topology. ....	25
1.16	Migration policy and extinctive selection.....	25
1.17	Reducing the feasible region $\phi = \{0.75, 0.5, 0.25\}$ . $K = 16$ , 10% migration, $m = 30$ , $n = 100$ .....	26
1.18	Increasing the number of constraints $m = \{5, 10, 30\}$ . $K = 16$ , 10% migration, $\phi = 0.25$ , $n = 100$ .....	26

XVIII List of Figures

1.19	Increasing the search space $2^n$ , $n = \{100, 250, 500\}$ . $K = 16$ , 10% migration, $\phi = 0.25$ , $m = 30$ . . . . .	26
1.20	Subpopulation size $\lambda$ , $K$ (10% migration, $m = 30$ , $n = 100$ , $\phi = 0.25$ ) . . . . .	27
1.21	Migration Rate $\lambda_m/\lambda$ ( $K = 16$ , $m = 30$ , $n = 100$ , $\phi = 0.25$ ) . . . .	27
1.22	Effect of Extinctive Selection ( $K = 16$ , $m = 30$ , $n = 100$ , $\phi = 0.25$ ) . . . . .	28
1.23	Fitness Transition ( $K = 16$ , $m = 30$ , $n = 100$ , $\phi = 0.25$ , $M = 20$ )	28
2.1	Pseudocode of a parallel EA . . . . .	34
2.2	Formulation and Pareto front of the problem Bihn2 . . . . .	36
2.3	Different models of parallel EAs . . . . .	37
2.4	Publications on parallel MOEAs . . . . .	40
2.5	Number of publications for each type of parallel MOEAs . . . . .	40
2.6	Application domains of parallel MOEAs . . . . .	41
2.7	Pseudocode for (1 + 1)-PAES . . . . .	43
2.8	CRND problem instance used . . . . .	46
3.1	Overall architecture of the hardware genetic algorithm proposed	62
3.2	Pseudorandom bitstream generators – Fibonacci vs. Galois implementation . . . . .	63
3.3	The architecture of the selection component . . . . .	64
3.4	The state transition function of the selection component controller . . . . .	65
3.5	The architecture of the crossover component . . . . .	66
3.6	The architecture of the crossover component . . . . .	66
3.7	Stochastic bipolar neuron architecture ([9]) . . . . .	67
3.8	Plotting Michalewics’s function ([8]) . . . . .	68
4.1	The RC1000 prototyping platform for reconfigurable computing	73
4.2	The RC1000 scheme showing its main components . . . . .	74
4.3	Visual C++ application implementing the GA for the TSP . . . .	75
4.4	Genetic algorithm used . . . . .	76
4.5	(a) Example of instance. (b) Optimal solution for this instance .	80
4.6	Graphical representation of the ratio between the HW/SW average execution times . . . . .	84
4.7	RLS identification of benchmark ball with different values for $\lambda$ .	86
4.8	PERLS behaviour . . . . .	87
4.9	PERLS architecture . . . . .	88
4.10	PERLS implementation in reconfigurable hardware . . . . .	90
5.1	Graphical representation of DNA sequencing and assembly [7] .	100
5.2	Schemes of two GA types along with a panmictic GA . . . . .	102
5.3	Graphical representation of the mean execution ( $\mu\text{sec}$ ) of a dGA depending on migration rate and gap . . . . .	108
5.4	Weak (versus panmixia) speedup of a dGA depending on migration rate and gap . . . . .	110
5.5	Weak (orthodox) speedup of a dGA depending on migration rate and gap . . . . .	111



5.6	Summary of the results of a dGA having two and four processors	113
6.1	The parallel island model	121
6.2	The parallel evaluation of a population	124
6.3	The parallel evaluation of a single solution	126
6.4	A layered architecture of ParadisEO-CMW	129
7.1	The 3D rendering pipeline	137
7.2	Addition of two matrices on GPU	138
7.3	Representing individuals of 32 variables on textures	139
7.4	The two fragment shaders for mutation process	142
7.5	The shader for fitness evaluation	144
7.6	Running example of median picking algorithm	146
7.7	Fitness value of the best solution found by the GPU and CPU approaches for functions $f_1$ and $f_2$	148
7.8	Fitness value of the best solution found by the GPU and CPU approaches for functions $f_3$ and $f_4$	148
7.9	Fitness value of the best solution found by the GPU and CPU approaches for function $f_5$	149
7.10	Execution time of the GPU and CPU approaches for functions $f_1$ and $f_2$	149
7.11	Execution time of the GPU and CPU approaches for functions $f_3$ and $f_4$	150
7.12	Execution time of the GPU and CPU approaches for function $f_5$	150
7.13	Fitness time graphs for functions $f_1$ and $f_2$	152
7.14	Fitness time graphs for functions $f_3$ and $f_4$	152
7.15	Fitness time graphs for function $f_5$	153
7.16	Four snapshots of genome-convergence visualization	153
7.17	Four snapshots of fitness-convergence visualization	153
8.1	The hybridization of soft computing	160
8.2	Object function $F$	163
8.3	Progress of $PSO$ on object function $F$	163
8.4	The distribution of particles at different iterations	164
8.5	Task and data distribution of pipeline processing	165
8.6	Task and data distribution of data parallelism	165
8.7	Communication Strategy 1 for Loosely Correlated Parameters	168
8.8	Communication Strategy for strongly correlated parameters	169
8.9	Migrate best particle between each pair group	169
8.10	Migrate the best particle according to the ring structure	170
8.11	A General Communication Strategy 3 for Unknown Correlation Between Parameters	171
9.1	Two amino acids forming a peptide chain	179
9.2	A sample protein conformation in the 2D HP Model. The black squares represent Hydrophobic (H) residues. Dashed lines indicate the non-adjacent H residue contacts. The 1 indicates a terminating residue	181

XX List of Figures

9.3	A sample protein conformation in the 3D HP Model. The black squares represent Hydrophobic (H) residues. Dashed lines indicate the non-adjacent H residue contacts. The 1 indicates a terminating residue . . . . .	181
9.4	Ants moving from nest to food todo – more explanation! . . . . .	184
9.5	Relative direction system for the candidate solution . . . . .	187
9.6	Speed differences between applications . . . . .	192
9.7	Trends in the MCOPE algorithm . . . . .	192
9.8	Computation time per iteration versus protein length . . . . .	193
9.9	Number of iterations versus protein length . . . . .	194
9.10	Network traffic vs protein length for client . . . . .	195
9.11	Network traffic vs protein length for server . . . . .	196

---

## List of Tables

1.1	7 subclasses of problems, 10 random problems in each subclass .	14
1.2	Genetic algorithms parameters. ....	15
1.3	Convergence Reliability of Conventional and Parallel Varying Mutation GAs with Deterministic Mutation Rates .....	20
1.4	Convergence Reliability of Conventional and Parallel Varying Mutation GAs Self-Adaptive Mutation Rates .....	20
2.1	Parallel MOEAs .....	39
2.2	Parallel algorithms for solving MOPs .....	42
2.3	Formulation of the multiobjective problems Fonseca and Kursawe	45
2.4	Entropy and contribution metrics of the three considered MOPs	48
3.1	Comparison of the performance results: software genetic algorithms (SGA), hardware engine for genetic algorithms (HEGA) and proposed hardware genetic algorithms (PHGA). (The area is expressed in terms of CLBs and the time is in seconds.) .....	68
4.1	Ratio between the HW/SW average execution times .....	80
4.2	Differences in the HW-time/SW-time ratio between a version and the previous one.....	81
4.3	Resource use, frequencies and periods for the different hardware versions .....	83
4.4	PERLS nomenclature: the more important algorithm's parameters .....	86
4.5	Timing results of the Evolutionary Central Unit.....	89
5.1	Parameter settings .....	106
5.2	Hardware and software environments .....	106
5.3	Mean execution time ( $\mu\text{sec}$ ) of a dGA having 8 processors over a Fast Ethernet network.....	107
5.4	Mean execution time ( $\mu\text{sec}$ ) of a dGA having 8 processors over a Gigabit Ethernet network.....	107
5.5	Mean execution time ( $\mu\text{sec}$ ) of a dGA having 8 processors over a Myrinet network.....	107

XXII List of Tables

5.6	Ratio between the communication time and the total runtime for a Fast Ethernet network . . . . .	109
5.7	Ratio between the communication time and the total runtime for a Gigabit Ethernet network . . . . .	109
5.8	Ratio between the communication time and the total runtime for a Myrinet network . . . . .	109
7.1	The Fast Evolutionary Programming Algorithm . . . . .	141
7.2	The set of test functions. . . . .	147
7.3	The ratios of the average execution time of the GPU (CPU) approach with different population sizes to that with population size of 400 . . . . .	151
7.4	Experimental result summary of $f_5$ . . . . .	151
7.5	The speed-up of the GPU approach . . . . .	151
8.1	Asymmetric initialization ranges and $V_{max}$ values . . . . .	172
8.2	Performance Comparison of <i>PSO</i> and <i>PPSO</i> with The First Communication Strategy for Rosenbrock Function . . . . .	173
8.3	Performance Comparison of <i>PSO</i> and <i>PPSO</i> with The First Communication Strategy for Rastrigin Function . . . . .	173
8.4	Performance Comparison of <i>PSO</i> and <i>PPSO</i> with The Second Communication Strategy for Griewank Function . . . . .	173
8.5	Performance Comparison of <i>PSO</i> and <i>PPSO</i> with The Third Communication Strategy . . . . .	174
9.1	Test sequences . . . . .	191
9.2	Emperical results . . . . .	193

---

## List of Algorithms

6.1	EA pseudo-code	119
8.1	Parallel Particle Swarm Optimization Algorithm	170
9.1	Single process ant colony algorithm	183
9.2	ACO construction phase	186
9.3	ACO local search	188
9.4	DSC server algorithm	188
9.5	MCOCE server algorithm	189
9.6	MCOPE server algorithm	190